Improving the Latency of 802.11 hand-offs using Neighbor Graphs

Minho Shin, Arunesh Mishra, William A. Arbaugh

{mhshin, arunesh, waa}@cs.umd.edu
Department of Computer Science
University of Maryland
College Park, Maryland 20742, USA

***Abstract*— The 802.11 IEEE Standard has enabled low cost and effective wireless LAN services (WLAN). With the sales and deployment of WLAN based networks exploding, many people believe that they will become the fourth generation cellular system (4G) or a major portion of it. However, the small cell size of WLAN networks creates frequent hand-offs for mobile users. If the latency of these hand-offs is high, as previous studies have shown, then the users of synchronous multimedia applications such as voice over IP (VoIP) will experience excessive jitter. The dominating factor in WLAN hand-offs has been shown to be the discovery of the candidate set of next access points. In this paper, we describe the use of a novel and efficient discovery method using *neighbor graphs* and *overlap graphs*. Our method reduces the total number probed channels as well as the total time spent waiting on each channel. Our implementation results show that this approach reduces the overall probe time significantly when compared to other approaches. Furthermore, simulation results show that the effectiveness of our method improves as the number of non-overlapping channels increases, such as in the 5 GHz band used by the IEEE 802.11a standard.**

I. Introduction

The IEEE Standard 802.11 [1] enables low cost and effective wireless LAN services. The unlicensed and free spectrum (2.4GHz in 802.11b/g and 5GHz in 802.11a) used by 802.11 networks permits the deployment of high speed (11Mbps in 802.11b and up to 54 Mbps for 802.11g/a) by organizations [2], and the major laptop and handheld computer vendors are quickly integrating WLANs devices into their equipment. This rapid adoption rate makes many people believe that 802.11 will become the fourth generation cellular system (4G) or a major portion of it. In fact, WLANs in public areas such as airports, hotels, universities [3] [4] and shopping centers [5] have already been successfully deployed. To meet this lofty goal, the hand-offs that occur when a user is mobile must be efficient.

The small cell size in WLAN creates frequent hand-offs potentially causing delays or disruption of communications if the latency of the hand-off is high. A major component of the hand-off process is identifying the new best available AP and associating to that AP (layer-2 hand-off). When IP connectivity is used, the additional process of layer-3 hand-off must also be completed [6].

Mobility and voice communications are the driving forces in current cellular networks, and the efficiency of both are paramount to the success of a new generation of cellular service. Due to the high bandwidth provided by 802.11 networks, Voice over IP (VoIP) is the logical choice for providing voice service. VoIP, however, requires maximum end-to-end delay of $50ms$ [7][8]. Unfortunately, previous studies have shown that majority of WLANs cannot complete layer- 2 hand-off process in 100 ms [9] [6] [10]. One study found that the observed layer 2 hand-off latencies are from $60ms$ to $400ms$ depending on the vendors of wireless cards and access points, and the study found that the probe phase (the discovery of next AP) is the dominating factor in layer 2 hand-off latency, accounting for more than 90% of the overall cost.

The probing latency is affected significantly by two parameters : probe count and probe-wait time[9]. In 802.11 networks, the probe count equals the number of channels to probe and the probe-wait time is a value between $MinChannelTime$ and $MaxchannelTime$ [1] ($7ms$ and $11ms$, respectively, in this paper). Since the IEEE Standard 802.11 does not specify a method for probing channels, wireless vendors use their own, usually proprietary, algorithms based on heuristics [9]. We categorize these algorithms as **Full-Scanning** and **Observed-Scanning**. Full-scanning is a brute force algorithm that probes all legitimate channels (11 channels in US [11]). Observed-scanning, on the other hand, limits probes to a subset of legitimate channels which have been previously observed [12]. The main benefit of observed-scanning over full-scanning is more easily seen by understanding how channel management usually occurs. In the U.S., there are three non-overlapping or independent channels (1, 6, and 11). The exclusive use of these channels prevents interference between adjacent channels. Thus in a network using only the non-overlapping channels, observed-scanning need only probe three channels instead of 11. Observed-scanning, however, does not work well when the number of non-overlapping channels is high as in 802.11a standard which has 12 non-overlapping channels.

In this paper, we propose two innovative and efficient layer-2 hand-off schemes, **NG algorithm** and **NG-pruning algorithm**. The NG algorithm uses a novel data structure called *neighbor graph* (NG) [13]. The NG-pruning algorithm further improves the discovery process by also using a novel data structure called *non-overlap graph*.

The neighbor graph is a data structure that abstracts the hand-off relationships between access points (AP). An access point, AP_1 has a handoff relationship with AP_2 if and only if a station can hand-off from AP_1 to AP_2. AP_2 is then said to be a neighbor of AP_1. The neighbor graph captures the following information :

1) The set of channels which neighbor APs are operating on.
2) The set of neighbor APs on each of these channels.

The non-overlap graph is a data structure that abstracts the non-overlapping relationships between APs. Two APs are non-overlapping if and only if a mobile station cannot communicate to both of them with acceptable link quality. When two APs are non-overlapping, a probe response from one of them indicates the unreachability to the other. Thus, during a hand-off, the station can exclude (prune) such unreachable APs from the list of APs to probe, resulting in a faster hand-off.

Using these data structures, the NG algorithm and NG-pruning algorithm can:

(i) reduce the number of channels to probe
(ii) reduce waiting time spent on each probed channel
(iii) (NG-pruning) reduce number of APs to probe.

We have implemented the probing algorithms (full-scanning, observed-scanning, NG and NG-pruning) and compare their performances by experiments in a deployed IEEE 802.11b indoor network. In our experiments, the NG algorithm reduces the probing latencies of full-scanning and observed-scanning by 80.7% and 30.8% on average, respectively. While the NG-pruning algorithm reduces the latencies of full-scanning and observed-scanning by 83.9% and 42.1% on average, respectively. Furthermore, simulations demonstrate the performance of the algorithms in various topologies using different parameters, such as the number of neighbors and the number of independent channels. The simulation results show that the performance gain of the NG and NG-pruning algorithms increases almost linearly as the number of channels increase, and the performance increases drastically as the average number of neighbors per channel decrease.

The paper is organized as the following. In section II, we provide background information on hand-offs and probing algorithms. Section III, IV and V describe and discuss the NG and NG-pruning algorithms. After discussing the results of our experiments and simulations in section VI and VII, related work is presented in section VIII. We conclude the paper in section IX.

II. BACKGROUND

In this section, we explain the basic cellular concept and the hand-off problems in cellular networks, leading to the discussion of hand-offs in 802.11 networks. The probing process is also described in detail.

A. Cellular Concept

The *Cellular Concept* was a major breakthrough in solving the problems of spectrum allocation and user capacity in the early mobile radio systems [14]. While a single, high powered base station with an antenna on a tall tower could provide a large coverage area, it made it impossible to reuse the same frequency throughout the area due to interference. One base station also imposed a serious limitation in user capacity. For example, the Bell mobile system in New York City in the 1970s could only support a maximum of twelve simultaneous calls over an area of a thousand square miles [14]. The cellular concept offered very high user capacity within a limited spectrum allocation by replacing a single large powered cell with many low powered cells, each covering only a small portion of the service area (Figure 1).

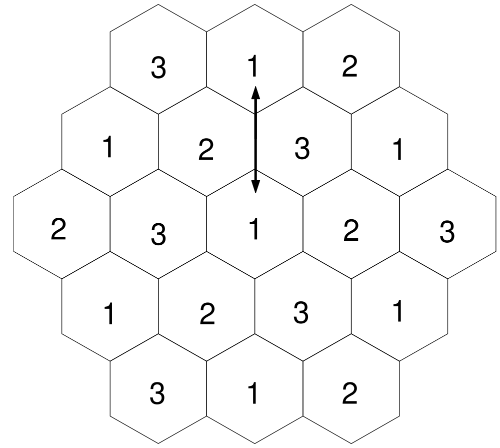

Fig. 1. *Cell concept and optimal channel allocations with three independent channels. The arrow shows the maximum distance between cells with the same channel.*

The cellular concept introduces two necessary techniques, *optimal channel allocation* and *hand-offs*. Optimal channel allocation allows non-adjacent cells to use the same channel while minimizing interference between such cells (see Fig. 1). Due to the decreased coverage of each cell, however, a mobile station may move into a different cell while a session is in progress. The hand-off process identifies the next base station (cell) and transfers ongoing session– ideally with minimal to no disruption of service.

In 802.11, to overcome the even smaller coverage area of an access point (on average 30 meters in 802.11b and smaller in 802.11a), multiple access points are necessary to cover the same area as a single CDMA or GSM base-station. The limited number of non-overlapping channels (three in 802.11b, twelve in 802.11a) also necessitate well designed channel assignments to avoid interference from neighboring access points. Non-overlapping channels, or independent channels, are channels that provides enough frequency separation to co-locate several radios links without interference. In 802.11b, two channels separated by more than 5 channels are known to be independent. We will discuss the effect of channel assignment to our hand-off scheme in a later section. Figure 1 shows an ideal channel assignment in 802.11b. Hand-offs are a major challenge in 802.11 networks since they occur more frequently due to the small coverage area of an access point. In Third Generation mobile systems[15], a mobile station can conduct seamless hand-offs by activating several radio links simultaneously (*Soft Handover* [16]). Unlike 3G mobile stations, a 802.11 mobile station must complete the hand-off process using only a single radio link (*Hard Handover*) as the standard currently prevents

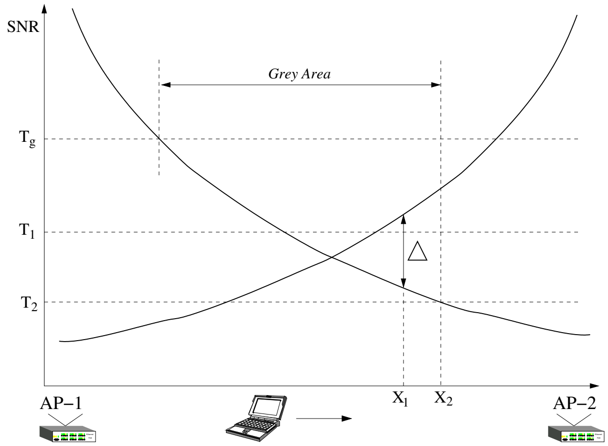

Fig. 2. *SNR changes between two access points and hand-off parameters.* Δ *denotes hysteresis. Station hand-offs at* X_1 *when* $T_h = T_1$ *and at* X_2 *when* $T_h = T_2$*. The shown Grey Area is when hand-off threshold* $T_h = T_2$*.*

a mobile station from associating to more than one access point within a network at the same time. This limitation makes efficient hand-offs difficult to achieve.

B. Hand-offs

In 802.11, a station leaving an access point is required to initiate a hand-off process for finding the next access point and establishing a link with that access point (reassociation). The stations must hand-off to maintain service continuity and load balancing of the system[10]. Properly designed hand-offs also should be done early to avoid interference with stations in other cells (cell dragging). To make the hand-off imperceptible to users, a fast hand-off is critical. For example, a hand-off completed in less than $50ms$ allows a VoIP user not only continuous conversation but also an unnoticeable transition of the call [8], [13]. However, a hand-off longer than $50ms$ can cause disruption of service due to the loss of more than two voice packets. A poorly designed hand-off can also incur *ping-pong hand-offs* due to the momentary fading of signal strength [17]. Therefore, when and how to hand-off is a very important design challenge.

When to initiate hand-off has been studied previously [17] [18] [19]. A hand-off algorithm called the *relative signal strength with hysteresis and threshold*, used by Lucent [12] is described below. Figure 2 depicts a typical *signal-to-noise ratio*[1] (SNR [20]) changes between two adjacent access points, AP_1 and AP_2. As the station moves from AP_1 to AP_2, the SNR from AP_1 decreases while SNR from AP_2 increases. $S_1(x)$ and $S_2(x)$ denote the SNR values from AP_1 and AP_2, respectively, at position x. Hand-off initiation is based on two parameters, T_h and Δ, both positive. At any position x, the station initiates a hand-off from AP_1 to AP_2 if and only if the following conditions hold :

$$\begin{cases} S_1(x) < T_h \\ S_2(x) - S_1(x) > \Delta \end{cases} \tag{1}$$

T_h is a threshold for the hand-off and Δ is a hysteresis. In Figure 2, the station triggers the hand-off process at position X_1 if $T_h = T_1$ and X_2 if $T_h = T_2$. The threshold condition avoids unnecessary hand-offs when the current link quality is sufficient, and the hysteresis condition avoids the ping-pong effect [17].

As soon as the hand-off condition holds, the station initiates a hand-off procedure. The following summarizes the steps required for the successful completion of a hand-off.

1. *Probing :* Discover the best available AP in the vicinity.
2. *Layer-2 Authentication :* Two kinds of authentications are provided : open authentication and WEP. WEP, however, is known to be insecure [21].
3. *Reassociation :* Establish the communication link with the found AP. Exchange necessary information such as supported transmission rates and beacon interval.
4. *802.11i Authentication :* Authenticate the user by 802.1x [22] and EAP-TLS [23] as described in IEEE 802.11i Standard [24] [25].
5. *Layer-3 Hand-off :* Update binding information and the care of address [26]. This also includes packet forwarding to minimize packet loss.

C. Probing in 802.11

Probing is the dominating factor in hand-off latency, accounting for more than 90% of the overall latency. The probing process (or scanning process) finds a new available AP with the best signal quality with respect to the station. Fig. 3 illustrates the probing procedure as described in the IEEE Standard 802.11 [1]. In the figure, N distinct channels are selected to probe. Once the channels to be probed are determined, the station switches to each selected channel and broadcasts a *probe request* frame. We call this the Channel Switch and Transmission overhead (CS&T) latency. In Fig. 3, the arrows toward APs represent such probe request frames broadcast on a channel (numbered in a circle). Upon receiving a probe request, an AP responds with a *probe response* frame to the station (downward arrows in Fig 3). After the transmission of the probe request, the station waits a certain amount of time (*probe-wait time*) before switching to the next channel. After probing all selected channels, the next access point is determined from the information received in the probe responses and their associated signal strengths.

The algorithm 1 describes the process described above. If the given channels to probe include all legitimate channels, the algorithm is called the full-scanning algorithm. If the channels are observed from previous probes (or possibly passive monitoring), it is called observed-scanning algorithm. Note that $MaxChannelTime \geq MinChannelTime$.

In line 4, the station determines whether to stop on $MinChannelTime$ or stay until $MaxChannelTime$. If the medium is idle for $MinChannelTime$, the station can conclude to stop and probe the next channel. However, if medium is detected to be busy before $MinChannelTime$ expires, the received packet can be either a probe response or other packets. Despite the second case, there is a good reason for the probing station to remain on channel because it is possible that an AP responding with a probe response fails to gain access to the medium due to contention with other transmitters.

[1]Other metrics may be used for hand-off decision, such as Received Signal Strength Indicator (RSSI), Bit Error Rate (BER) or Signal-to-Interference Ratio (SIR). We select SNR in our implementation.

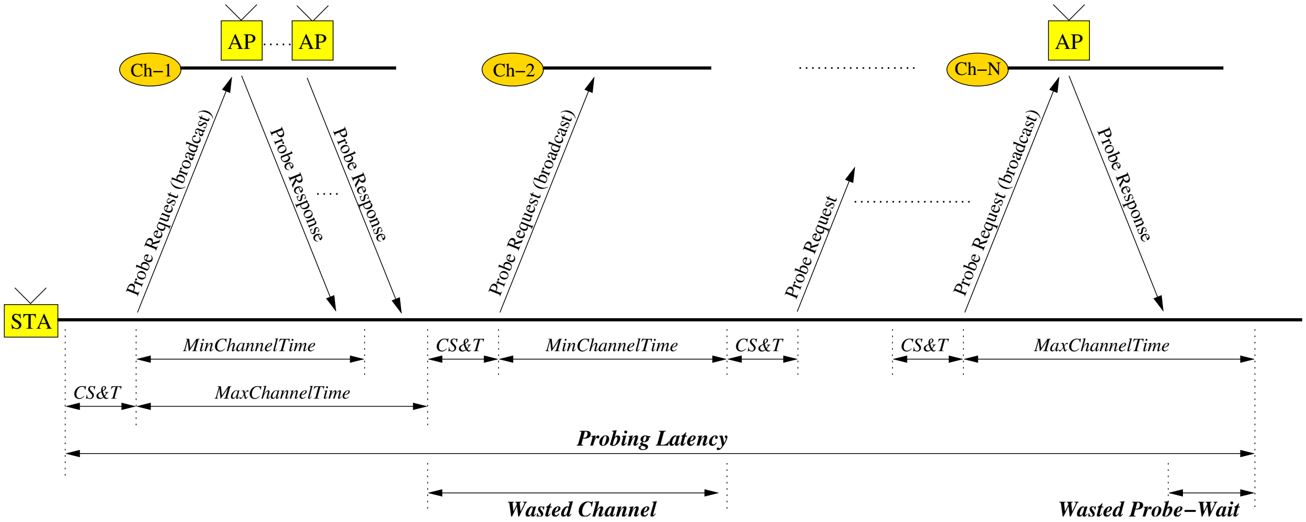

Fig. 3. *A Probing Process in IEEE Standard 802.11. CS&T in the figure means "Channel Switching and Transmission Overhead".*

Algorithm 1 Full-scanning or Observed-scanning algorithm

```
for each channel to probe do
    Broadcast probe request on this channel
    Start probe timer
    if Medium is idle until MinChannelTime expires then
        continue
    end if
    if MaxChannelTime expires then
        continue
    end if
end for
```

Fig. 3 illustrates both cases when the probe-wait time is $MinChannelTime$ or $MaxChannelTime$. In the figure, the station starts probing by switching to channel *Ch-1* and transmitting a probe request frame. Since the medium is detected busy before $MinChannelTime$ expires, the station keeps waiting until $MaxChannelTime$ expires. On channel *Ch-2*, the medium is detected to be idle until $MinChannelTime$ expires. Therefore, the station stops waiting and proceeds to next channel and so on. Note that this algorithm is wasting time on channel *Ch-2* where no APs exist (wasted channel). Moreover, on channel *Ch-N*, the algorithm keeps waiting for more arrivals even after receiving the last probe response (wasted probe-wait). We claim that the removal of wasted channels and wasted probe-wait time is possible with prior knowledge of local topology, presented by neighbor graph.

III. Neighbor Graph[2] and Non-Overlap Graph

In this section, we re-define neighbor graph [13] in a slightly different fashion : aggregation of personal, *i.e.* the neighbor graph is generated from the perspective of the mobile station, neighbor graphs with directed edges, and we discuss the properties that affect the performance of our proposed probing algorithms. We also define the notion of a non-overlap graph.

A. Definitions

Assume $\mathbf{AP} = \{AP_1, AP_2, \ldots, AP_n\}$, is a set of APs in a WLAN under consideration. The association pattern (or mobility pattern) of a station c during a finite period of time can be denoted as $\Gamma(c) =< AP_{c1}, AP_{c2}, \ldots, AP_{ck} >$ where k is the number of access points that served the station and AP_{ci} is the ith such access point. For simplicity, assume that each transition to the next access point is done by reassociation[3]. From the sequence $\Gamma(c)$, we can construct a *personal neighbor graph* for station c, $PG(c) =< V, E >$ where $V = \mathbf{AP}$ and

$$E = \{< AP_i, AP_j > | \, AP_i \text{ and } AP_j \text{ are successive in } \Gamma(c)\}.$$

By the *Locality Mobility Principle* in [13], these edges reflect the hand-off relationships between APs. Personal neighbor graphs are directed graphs that reflect the mobility patterns of stations in that WLAN environment.

Now we define a *Neighbor Graph*. Let $\mathbf{C}$ be a set of stations in the network and $PG(c)$ be the personal neighbor graph of station $c \in \mathbf{C}$. Then, the neighbor graph NG is defined as

$$NG = \bigcup_{c \in \mathbf{C}} PG(c)$$

.

B. Asymmetry of Hand-offs

A hand-off relationship reflects the geographical closeness of two APs to each other. One would expect a hand-off relation to be symmetric, however in reality, the hand-offs are often asymmetric, i.e, hand-off relationship from AP_i to AP_j does not

[2]Proactive caching with neighbor graph has been added to IEEE Standard 802.11f [27] [13].

[3]Some of the transitions can be associations, *i.e*, without a hand-off. For example, the station can shut-down and move to another cell, or the link to the network has been dropped due to non-overlapping coverage of two access points. Note that the consecutiveness of reassociations does not affect the definition and the generation of neighbor graphs.

imply the hand-off relationship from AP_j to AP_i. The asymmetric nature of the hand-off relationship is due to geographical characteristics, irregularity of radio coverage, high AP density, or user's uni-directional mobility patterns. In our experiments, we observed that 57% of hand-off relationships are asymmetric (see Fig. 4).

C. Degree in Neighbor Graph

The degree of an AP in a neighbor graph is the number of outward edges from that AP. This represents the number of neighbors to which a station can hand-off. With a sufficient number of independent channels such as in 802.11a, the degree of an AP bounds the number of probings in NG and NG-Pruning algorithms, affecting the performance of the algorithms. The discussion below shows that in most deployments, the degree of the neighbor graph is bounded by 6.

Despite the irregular contour of radio coverage in reality, a circular model is sometimes drawn as the shape of a cell with one access point [4]. However, the hexagon has been adopted as the most appropriate notation of cells in the literature [28]. Fig.1 shows a typical representation of a wireless service area divided by multiple cells. There are several reasons to use the hexagon as a geometric model of a cell. For the simplicity of analysis, non-overlapping regular congruent polygons, *i.e.*, polygon with same length of sides, are preferred over than circles. Covering a Euclidean plane with regular congruent polygons is called *Regular Tessellation*. There are only three regular polygons that tessellate the plane : triangles, squares and hexagons. Among these polygons, hexagon covers the area with fewest number of cells and also closely approximates a circular shape [14]. Under the hexagon tessellation model, the maximum number of neighbor cells is six in a 2D-plane. In reality, the maximum degree of a neighbor graph tends to be smaller in buildings due to the structure of the building and restrictions on mobility due to walls, etc.. In our experiment, the average number of neighbors was measured as 3.15.

When the network must co-locate multiple access points to increase user capacity, the hand-off process must consider load balancing among APs [10]. A neighbor graph with knowledge of neighboring loads can be used for load balancing. In this case, a neighbor count of more than six is possible.

D. Quality of Neighbor Graph

In this section, we introduce two metrics that reflect the quality of neighbor graphs. A neighbor graph is an approximation of an actual mobility graph for users. Given a set of m users $U = \{u_1, u_2, \ldots, u_m\}$, denote the aggregation of the actual mobility graphs of users U as M_U. M_U is a directed graph such that $M_U = (V, E_M)$ where E_M is the set of edges and V is the set of APs. Due to changes in topology or in user's mobility patterns, M_U may dynamically change over time, degrading the accuracy of the present neighbor graph. In the following definitions, we assume that N is a neighbor graph such that $N = (V, E_N)$ where E_N is the set of edges and V is the set of APs.

[4]Some antennas, such as sector antenna, can produce non-circular coverage. This paper assumes omni-directional antenna is used

1) Error of neighbor graph: The error of a neighbor graph is defined as the ratio of the number of missing edges to the number of edges in the true mobility graph, i.e,

$$Er(N, M_U) \stackrel{\text{def}}{=} \frac{|E_M - E_N|}{|E_M|} \tag{2}$$

which takes a value in $[0, 1]$. If $Er(N, M_U) = 0$, neighbor graph N is *valid*. The error of a neighbor graph also represents the probability that a station will hand-off to an AP which is not a neighbor in the graph. With an error of non-zero, NG fails to provide the same quality of service with other methods without NG. For example, with some missing edges, NG probing fails to discover the best access point as full-scanning or observed-scanning. Therefore, maintaining NG as close as possible to a valid graph is important.

Any superset of a mobility graph (a complete graph, for instance) is a valid neighbor graph. We claim that the neighbor graph generated and maintained by the methods in section V creates a close approximation of a valid graph, and the generated converges to a valid graph by self-adaptations with minimal cost.

2) Overhead of neighbor graph: The overhead is another metric that indicates the quality of a neighbor graph. We define the overhead of a neighbor graph as the ratio of redundant edges to the number of edges in actual mobility graph, i.e,

$$Ov(N, M_U) \stackrel{\text{def}}{=} \frac{|E_N - E_M|}{|E_N|} \tag{3}$$

which takes a value in $[0, 1]$. If $Ov(N, M_U) = 0$, neighbor graph N is *compact*. The overhead of a neighbor graph also represents the probability that an edge in the neighbor graph does not represent a reassociation relation.

Even with a valid neighbor graph, redundant edges can impair the performance of an algorithm using the neighbor graph. The generation method suggested in section V always converges to a compact neighbor graph by discarding obsolete edges by a timeout.

E. Non-Overlap Graph

In this section, we introduce a novel conceptual graph, the *Non-Overlap Graph*. The overlap graph (OG) is an undirected graph over APs, the set of all access points in the network. An edge of the overlap graph, $< AP_i, AP_j >$ represents an overlapping relationship between APs. AP_i and AP_j *overlap* if there exists a location where a station can communicate to both of them with "acceptable" link quality. Acceptable link quality means a link quality good enough to avoid the hand-off being triggered (negation of equation **??** in section II). Such access points are called *reachable* from the station and otherwise, called *unreachable*. Note that a neighbor graph is a subset of overlap graph, i.e,

$$NG \subseteq OG. \tag{4}$$

When the edges of an overlap graph are considered bi-directional edges, the overlap graph is a valid neighbor graph, i.e,

$$Er(OG, M_U) = 0 \tag{5}$$

where M_U is the actual mobility graph of users U.

But OG is not necessarily compact ($Ov(OG, M_U) \geq 0$).

A *Non-Overlap Graph* (NOG) is an undirected graph that is the complement of the overlap graph, meaning that $< AP_i, AP_j >$ is an edge in Non-overlap graph if and only if $< AP_i, AP_j >$ is NOT an edge of overlap graph. That is,

$$NOG \stackrel{\text{def}}{=} OG^c. \tag{6}$$

- **Principle of non-overlapping** : The reachability to one of the non-overlapping APs implies the unreachability to the other AP.

This principle enables the "pruning" algorithm described in the following section.

IV. New Probing Algorithms

A. NG Probing

There are two main factors that affect probing latency:

(i) the number of channels to probe, *probe-channel count*
(ii) waiting time on each probed channel, *probe-wait time* [9].

With prior knowledge of the neighbor graph, the probe-channel count and probe-wait time can be reduced. The algorithm 2 describes the NG algorithm.

Algorithm 2 : NG algorithm

1: $\mathbf{P}$ = $\{P_i\}$, a partition of neighbors,
  where all APs in P_i have channel i
2: **while** $\mathbf{P} \neq \emptyset$ **do**
3: pick P_i from $\mathbf{P}$
4: $\mathbf{P} = \mathbf{P} - P_i$
5: Broadcast *probe request* on channel i
6: Start probe timer
7: **if** Medium is idle until MinChannelTime expires **then**
8: **continue**
9: **else if** all access points in P_i reply **then**
10: **continue**
11: **else if** MaxChannelTime expires **then**
12: **continue**
13: **end if**
14: **end while**

In line 1, the algorithm constructs $\mathbf{P}$, a partition of neighbor APs based on their channel assignments. Let the neighbor channel count be the number of distinct channels used by neighbor access points. Using $\mathbf{P}$, the algorithm probes only non-empty (AP existing) channels, removing wasted channels, or empty channels, in Fig.3. We call this process *minimum-channel probing*. In line 9, the algorithm shortens the probe-wait time to avoid wasted probe-wait time in Fig.3 called *optimal-wait probing*.

Note that the neighbor channel count is always less than or equal to the number of independent channels. As seen in Fig.1, an optimal channel assignment will avoid assigning the same channel to neighbors. In this case, neighbor-channel count is strictly less than the number of independent channels. The difference grows as the number of independent channels increases (802.11a), and our simulation results show that the benefit of minimum channel probing increases as the number of independent channels increase.

B. NG-Pruning algorithm

We use the non-overlap graph to gain additional performance improvement in probing. With prior knowledge of the non-overlap graph, the station prunes all of the non-overlapping APs from reachable APs. For example, assume that AP_i and AP_j are non-overlapping. Once the station gets probe response from AP_i, by the principle of non-overlapping, it is impossible to receive a probe response from AP_j with acceptable link quality. Thus, there is no reason to expect any response from AP_j (pruned). If the station cannot receive any probe response from AP_i with acceptable link quality, then we cannot drive any conclusion about the reachability of AP_j. The algorithm 3 describes NG-Pruning algorithm.

Algorithm 3 : NG-Pruning algorithm

1: $\mathbf{P}$ = $\{P_i\}$, a partition of neighbors,
  where all APs in P_i have channel i
2: **while** If $\mathbf{P} \neq \emptyset$ **do**
3: pick P_i from $\mathbf{P}$ with maximum degree in NOG
4: $\mathbf{P} = \mathbf{P} - P_i$
5: Broadcast *probe request* on channel i
6: **if** Medium is idle until MinChannelTime expires **then**
7: **continue**
8: **end if**
9: **for all** *probe response* from AP_r **do**
10: $P_i = P_i - \{AP_r\}$
11: $D = \{AP \mid (AP_r, AP) \in NOG\}$
  /* set of non-overlapping APs with AP_r */
12: $P_i = P_i - D$
13: **for all** $P_j \in \mathbf{P}$ **do**
14: $P_j = P_j - D$
15: **end for**
16: **if** $P_i = \emptyset$ or $MaxChannelTimes$ has expired **then**
17: **break**
18: **end if**
19: **end for**
20: **end while**

The algorithm above virtually creates a *local non-overlap graph*, a subset of non-overlap graph comprised of only neighbor APs. In line 3, the degree is examined in a local non-overlap graph. Line 11 $\sim$ 12 prunes any non-overlapping APs from P_i. By also removing non-overlapping APs from $\mathbf{P}$ (line 13 $\sim$ 15), we exclude non-overlapping APs from not probed channels. Such pruning reduces the number of APs to probe and sometimes resulting in the exclusion of a channel to probe.

V. Generation and Maintenence of NG

It is a very difficult process to create a neighbor or overlap graph by an analytical or manual method due to the irregularity

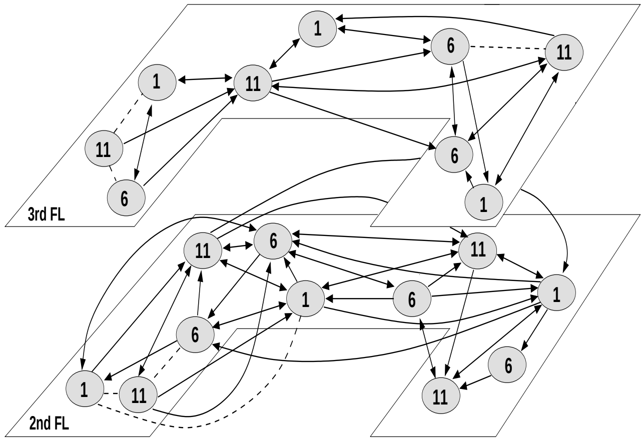

Fig. 4. *Generated Neighbor Graph (solid arrows) and Overlap Graph (dashed lines)*

of radio coverage. For example, a station in the line of sight to an access point can maintain good link quality to that AP up to a point. But a station closer to the same AP often cannot obtain link quality as good as the more distant station because of obstacles between station and AP. This unpredictability of radio coverage makes *empirical methods* practical and reliable for the generation of neighbor graphs and overlap graphs.

In this section, we discuss empirical methods for the generation of overlap graph and suggest a method for NG generation using overlap graph. We also discuss the freshness of neighbor graphs.

A. Generation of Overlap Graphs

To generate an overlap graph, the stations must measure the overlaps of cells and report to the system. When

$$T_h < S_i(c) < T_g$$

, i.e, the station c is in a gray area of AP_i (see Fig.2), with probability p, the station initiates a full-scanning probe and reports the overlapping access points to the system, called *overlap test*.

The gray area is where the probability of overlap with other APs is high. This excludes unnecessary overlap tests where the station is too close to the current AP. Excluding the non-gray area from overlap tests does not miss any overlap relationships due to the continuity of radio coverage. The overlap test probability, p, must be high in the initial phase for fast convergence and decrease over time to a positive minimum value. An overlap test creates only minimal reporting overhead on the stations.

A system can have a dedicated sampling phase in which a sampling station roams throughout the network, building the overlap graph to expedite the process.

B. Generation of Neighbor Graph

[13] proposes methods for NG generation. In the scheme, the system learns the edges of the NG from actual hand-offs of stations (*NG learning*) from either reassociation request frames or IAPP ([27]) Move-Notify messages. With a one time high cost probings (i.e, full-scanning), the first handoffs on each edge leaves a footprint as a new edge to neighbor graph. Over time, the neighbor graph will grow and converge to the actual mobility graph.

In this method, NG is always compact ($Ov(NG) = 0$) while error, $Er(NG)$, is large in the beginning and decreases to 0, becoming a valid graph. Therefore, due to the initial high error of the neighbor graph, NG cannot be enabled until NG becomes sufficiently close to valid neighbor graph. This will occur only in very early stages of a network.

To remove the possibility of a long convergence period, a system can alternatively have a dedicated NG generation phase like the one for the overlap gaph, in which a special station roams throughout the network adding as many edges as possible to the neighbor graph. Once all edges are added, the station can cease operation.

To reduce the NG generation period, we propose to exploit generated overlap graph as an initial neighbor graph. The generation of overlap graph can be completed much faster than the neighbor graph because overlap test does not require any mobility of the stations. For fast generation of the overlap graph in initial phase, an overlap test probability close to one can be adopted. Noting that the OG is a superset of the NG in equation 4, OG is a valid but not compact neighbor graph, that is, OG does not have any missing edges but does have some redundant edges. Such redundant edges are a pair of APs that overlap each other but no handoff is possible, called overlap-only edges. In our experiments, we find overlap-only edges are 12% of all overlap edges, and 35% if inter-floor edges are included. These redundant edges will be discarded over time by timeout. Once a station hand-offs along an edge in OG, the system adds that edge to a confirmed neighbor graph. In this way, the overhead of the initial NG will decrease to zero.

C. Freshness of Neighbor Graphs and Overlap Graphs

For freshness of neighbor graph, [13] suggests using a timestamp based LRU approach. In this section, we propose additional maintenance methods to keep neighbor graph adaptive to changes of both mobility patterns and AP topology.

The changes in a user's mobility pattern degrades the accuracy of a generated neighbor graph, impairing the compactness and validity. For example, new hand-off edges can be introduced by new mobility patterns, decreasing the validity of the NG. Also obsolete hand-off edges can harm the compactness incuring a degradation of performance. The timestamp based LRU method eliminates outlying edges caused by unusual mobility or by the change of mobility patterns. Eviction of rarely used edges increase the compactness, and thus increase the performance of NG. If a station follows a new mobility path that is not covered by the current NG, NG probing algorithm may fail to find a candiate AP. In such case, a one-time full scanning will successfully add a new adge, making NG valid once again.

NG also can adapt to additions and deletions (or failures) of APs in the network. When APs are deleted, all the edges to such APs will be evicted due to their inactivity. This leaves some redundant and isolated nodes in NG but with no harm on the performance. The addition of a new AP will be also detected by overlap tests. Once a new AP is found responding to the overlap tests, all other overlapping APs will add an temporary

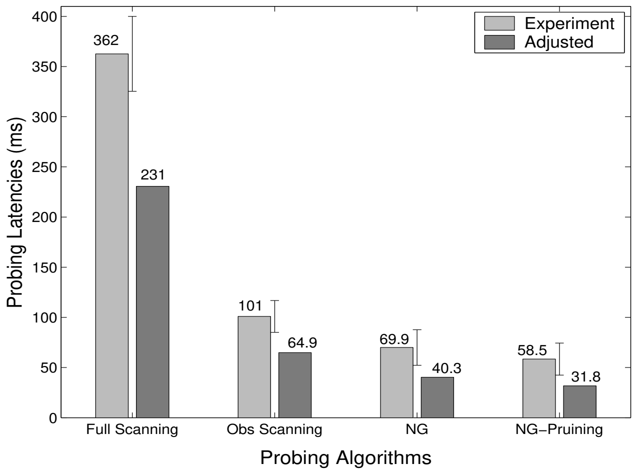

Fig. 5. *Probing Latencies with different Probing Algorithms, and Adjusted Probing Latencies with Channel Switch and Transmission overhead = 10ms. Confidence intervals are also shown.*

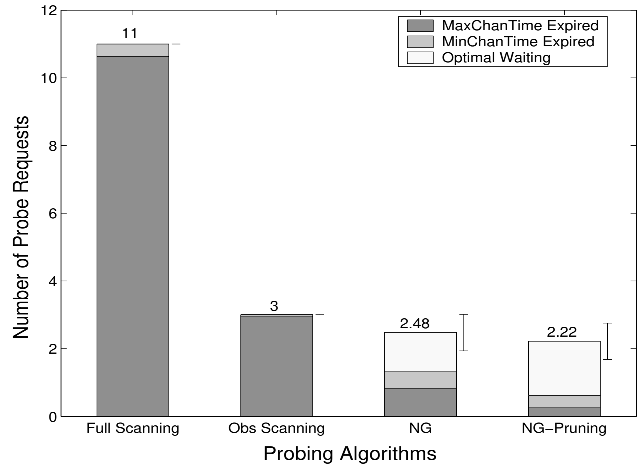

Fig. 6. *Probing Counts and its components with different Probing Algorithms. Each probing is a type of either MaxChannelTime expired, MinChannelTime expired or optimal waiting.*

TABLE I

SUMMARY OF EXPERIMENT RESULTS.

Algorithms	**Probe Count**	**Probe-Wait**	**Latency**
Full-Scan	11.0	10.9 ms	362 ms
Observed-Scan	3.0	11.0 ms	101 ms
NG	2.5	6.3 ms	70 ms
NG-pruning	2.2	4.4 ms	59 ms

edge to the new AP and run the timer to see if it is an actual NG edge.

VI. EXPERIMENTS

In this section, we discuss the implementation of the probing algorithms in a deployed 802.11b indoor network. We describe the network configurations, implementations, process of the experiments and the results. In brief, we implemented four different algorithms (full-scanning, observed-scanning, NG, NG-pruning) and measured approximately 250 hand-offs on the 2nd and 3rd floors in a campus building for each algorithm. The NG algorithm reduced the probing latencies of full-scanning and observed-scanning by 80.7% and 30.8%, respectively. While the NG-pruning algorithm reduced the latencies of full-scanning and observed-scanning by 83.9% and 42.1%, respectively. Table I summarizes the results.

In table I, probe count means how many probe request frames are broadcast, which equals to the number of channels probed. Note that NG and NG-prunning reduces both probe-wait time and probe count while observed-scanning only reduces probe count.

A. Experiment Configurations

The deployed wireless network spans two U-shaped floors in a campus building. There are nine APs on third floor and eleven APs on the 2nd floor. Each access point is a Cisco 350 with a one foot long *yagi* antenna on the ceiling. Open authentication is used for layer 2 authentication, and access points are assigned channels of 1, 6 and 11, which are known to be independent in 802.11b [11]. The geometry of the floors and topologies of the twenty access points are shown in figure 4. For the mobile station, a laptop with an Intel Pentium 4 Mobile 1.80 GHz and 256 MB RAM, equipped with a Prism 2.5 based Demarctech card [29] is used. Linux is used as an operating system for implementations and experiments.

B. Implementations of Algorithms

In most commercial wireless cards, the hand-off process is implemented inside the firmware for efficient operation. Because the firmware is considered proprietary by all vendors, we emulated the hand-off process using an open source WLAN driver in Linux, called *Airjack driver* v0.6.2-alpha [30]. We use Airjack to implement a roaming daemon in user space that emulates the hand-off process using the different algorithms. The daemon program, named *roamd*, cooperates with the customized Airjack driver so that the station, continuously monitoring the SNR with the current AP, initiates the hand-off process including probing, L2 authentication and reassociation when a certain hand-off condition holds. The Airjack driver is customized for the monitoring functionality.

The Airjack driver works with Prism 2 based wireless cards. As described in section II-C, the probing latency is affected by the cost of channel switching and the transmission latency. In our experiment, the latency for a channel switch and transmission is measured on average as 22.2ms (11.3ms for channel switch and 10.9ms for transmission). Compared to a 12.4ms, channel dwell time measured in [9], this 22.2ms[5] is too high for fast hand-offs. This latency is due to the performance limitations of either Airjack driver or the Prism chip itself.

For implementing the pruning algorithm, we use a generated overlap graph instead non-overlap graph due to implementation convenience.

[5]We have tested several Prism based wireless cards by different vendors with the same results.

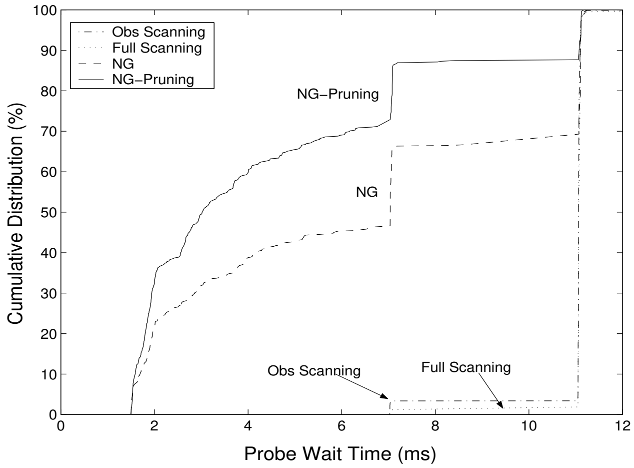

Fig. 7. *Cumulative distributions of probe-wait times illustrated for each algorithms.*

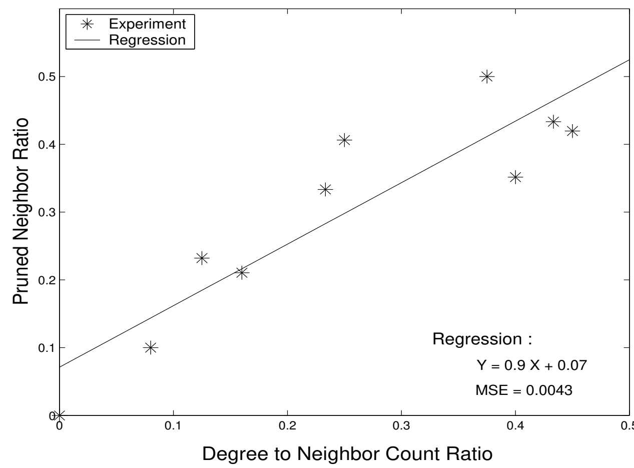

Fig. 8. *Pruning performance affected by degree to neighbor count ratio. MSE is Mean Square Error of the shown regression line.*

C. Measurement Methodology

Since the probing process is controlled by a roaming program, we measure the latencies inside the roaming program for the measurement. To overcome the clock resolution of $10ms$ in Linux, we patched the kernel to get microsecond resolution [31].

Sniffing is often used for performance analysis in WLAN, especially for management frames [9] [13] [32]. One advantage of sniffing is its independence from implementations. However, sniffing has limitations on measuring latencies inside the system. The actual probing latency begins with internal state transition from normal state to probing state, not with the end of successful transmission of first probe request frame. Sniffing is also involved with time synchronization problems to merge data sniffed from different machines. Our measurement takes the advantage of inside measurement in the roaming program. .

D. Experiment Process

Our experiments consists of the following three parts.

1. *Generation of neighbor graph*
2. *Generation of overlap graph*
3. *Measurement of probing latencies.*

For the generation of neighbor graph, we use the dedicated NG generation phase as described in section V-B. We first construct the neighbor graph inside the station by roaming throughout the 2nd and 3rd floors so as not to miss any possible reassociation edges. For the generation of the overlap graph, we issue 475 random overlap tests while roaming throughout the area. The station measuring the probing latencies is aware of the neighbor graph and overlap graph and makes use of them in probing algorithms. For measurement of probing latencies, we induce 250 hand-offs for each of the four different algorithms over all possible reassociation edges. For identical conditions across the algorithms, we follow exactly same path for each algorithm.

E. Experiment Results

1) Neighbor Graph and Overlap Graph: Fig. 4 shows the neighbor graph and overlap graph constructed by the generation phase. Each circle represents an access point with assigned channels labeled inside the circle. Arrows with a solid line represent an edge in both the neighbor graph and the overlap graph. The direction of an arrow shows the direction of the hand-off relationship. A dashed line represents an overlap-only edge in the overlap graph. We find that the number of neighbors is 3.15 on average with a maximum of 6 while the average neighbor channel count is 2.25.

2) Probing Latencies and Adjusted Latencies: The probing latencies measured are shown in Fig. 5. The x-axis shows the four different algorithms tested and the y-axis shows the probing latencies in milliseconds. The left bars are the results of our experiments while the right bars show adjusted probing latencies that one could achieve when switching channel and transmission overhead was $10ms$ as would be the case if the algorithm was implemented in firmware. From the experiments, the NG algorithm reduces the probing latencies of full-Scanning and observed-Scanning by 80.7% and 30.8%, respectively. Whereas the NG-pruning algorithm reduced the latencies of full-scanning and observed-scanning by 83.9% and 42.1%, respectively. The adjusted probing latencies show that fast probing can be less than $50ms$ when using our algorithms.

3) Probing Counts and Probe-Wait Time: The analysis of the probing counts and the probe-wait times are provided by Fig. 6 and 7. The NG and the NG-Pruning algorithm reduce probing counts of full-scanning by 77.45% and 79.82%, respectively while reducing observed-scanning by 17.33% and 26.00%, respectively. While not only reducing the probing counts, our algorithms minimize the number of MaxChannel-Time expirations and maximize the number of optimal-wait probings. In figure 6, more than 97% of probing accounts are MaxChannelTime expirations in both full-Scanning and observed-Scanning algorithms. However, the ratio of optimal

TABLE II
CONSTANTS USED IN SIMULATIONS

Constants	Values
$MaxChanTime$	11 ms
$MinChanTime$	7 ms
Round Trip Time (RTT)	2 ms
Channel Switch & Transmission	5 ms

TABLE III
VARYING PARAMETERS IN SIMULATIONS

Varying Parameters	Values
The Number of Neighbors	$\{2, 3, \ldots, 8\}$
The Number of Channels	$\{3, 5, 8, 12\}$

waiting constitutes 46.3% in NG, and 72.0% in NG-Pruning, resulting in improvements in latency. In the experiments, MaxChannelTime expires in $11ms$ and MinChannelTime in $7ms$ as recommanded in [9]. Optimal-wait probing is measured to be $2.7ms$ on average with standard deviation of $1.4ms$.

Fig.7 illustrates cumulative distributions of probe-wait time for each algorithms. The graph clearly shows that NG and NG-Pruning has much more shorter probe-wait times than the other two algorithms. For example, probings less than $7ms$ are 46.6% in NG and 71.2% in NG-Prunings. The leaps at $7ms$ and $11ms$ explains the high density around MinChannelTime ($7ms$) and MaxChannelTime ($11ms$).

4) Pruning performance vs the number of neighbors: NG-pruning outperforms NG algorithm by excluding some APs by using the non-overlap graph. The performance of the pruning algorithm over NG is illustrated in Fig.8. The x-axis is the ratio of degree to neighbor count and y-axis is the ratio of the number of pruned APs to neighbor count. In the pruning algorithm, only the local non-overlap graph is considered. The degree in the graph is the average number of non-overlap degrees in local non-overlap graph. The figure shows that the ratio of the degree to the number of neighbors directly affects the performance of the pruning algorithm, measured by the ratio of pruned AP with respect to the number of neighbors. We analyze the impact of the degree-to-neighbor-count ratio to the pruning performance in section VII.

VII. SIMULATIONS

The goal of simulation is to investigate the performances of compared algorithms under different environments from the experiment configurations. The main expansions of the parameters are :

- The number of independent channels up to 12
- The number of neighbors up to 8
- The channel assignments to be optimal

In an optimal channel assignment, no adjacent APs have the same channel, if possible. We only simulate local topologies, i.e, the current AP and its neighbors.

A. Simulation Model

The following assumptions are made for the simplicity of simulations.

1) The radio coverages of all APs are identical circles centered by the serving APs.
2) The positions of neighbors, $\{AP_1, AP_2, \ldots, AP_m\}$ where m is the number of neighbors, are randomly chosen around the current AP, AP_c with the following conditions :
$$\begin{cases} \text{For } i = 1, 2, \ldots, m, \\ \quad R \le Distance(AP_c, AP_i) \le 2 \times R \\ \\ \text{For distinct neighbors } AP_i \text{ and } AP_j, \\ \quad Distance(AP_i, AP_j) \ge R \end{cases}$$
where R is the radius of the coverage and $Distance$ is the euclidean distance between access points.
3) Access points AP_i and AP_j overlap each other if
$$Distance(AP_i, AP_j) \le 2 \times R$$
4) The direction of the station is randomly chosen so that there exists at least one neighbor AP to handoff.
5) AP_i is considered to be reachable by the station c if and only if
$$Distance(c, AP_i) \le R$$
6) There exists no contending other stations

The constant values are shown in table II

B. Simulation Process

Table III shows the varying parameters used across the simulations.

For each combination of parameters, ten different local topologies are randomly generated according to the model. Channels are assigned to the APs according to the following rules :

- The same channel with current AP is not assigned to any of neighbors.
- If $channel\ count > neighbor\ count$,
  assign distinct channels to neighbors while leaving one channel for current AP
- otherwise,
  assign ($channel\ count - 1$) channels so that no overlapping neighbors have the same channel. When $channel\ count = 3$, assigning the same channel to overlapping APs may be inevitable

Once a topology and channel assignment are determined, the station makes ten different handoffs toward randomly chosen directions and measure the probing latencies using four different algorithms. The figure 9 illustrates an example of generated topology when neighbor count is 5 and channel count is 4.

The dashed circle is current AP and others are its neighbors. The numbers represents assigned channels while alphabets are the names for reference purpose. The star and an arrow illustrates the mobility of a station. Note that only access points D and E are reachable to the station at the point of handoff. Note that neighbor cells does not need cover all the boundary of current AP.

The Fig. 10 shows the generated local non-overlap graph from above topology with average degree of 2.8.

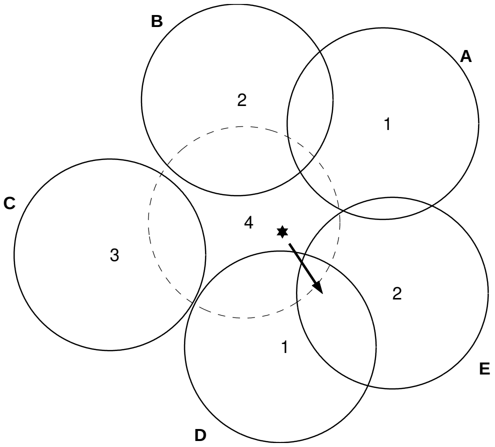

Fig. 9. Example of Topology Generated in Simulations. Dashed circle is current AP, solid circles are neighbor APs. Number in circle is assigned channels and alphabet for their names. The station, represented by a star is moving toward the direction of arrow.

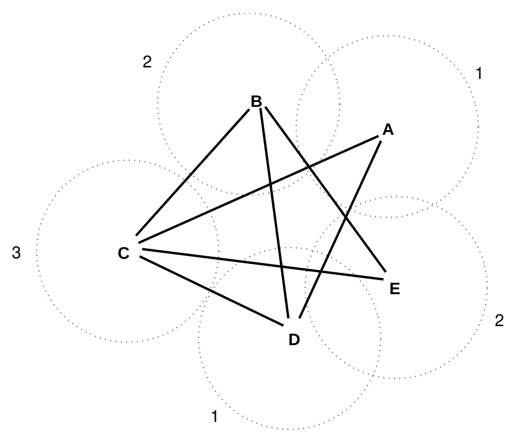

Fig. 10. Example of a Non-overlap Graph Generated in Simulations

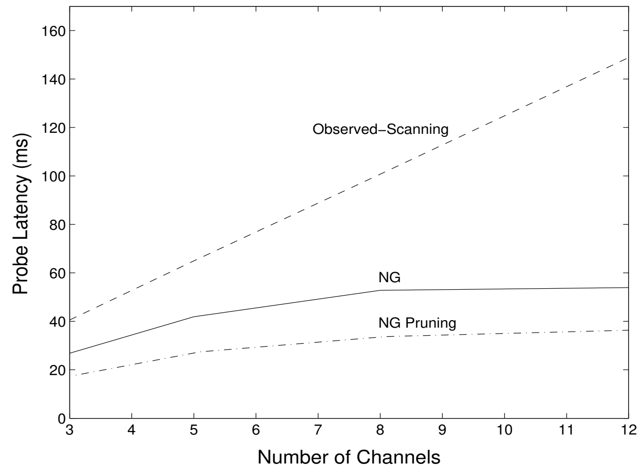

Fig. 11. *probing latencies of three algorithms by the number of channels*

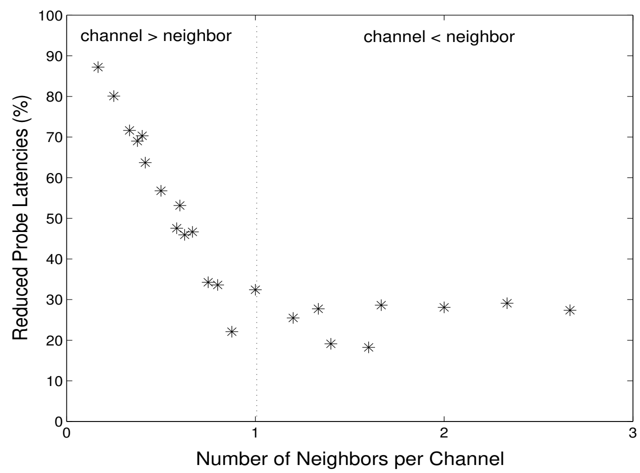

Fig. 12. *Performance of NG algorithm over observed-scanning, as a function of Neighbor-per-channel Density*

C. Simulation Results

1) Increasing Number of Channels Improves Performance: The Fig. 11 clearly shows that the performance of our algorithms improves with increasing number of independent channels. The table IV shows the reduced probing latencies of observed-scanning algorithm for different channel counts.

TABLE IV

Channel Counts	NG	NG-Pruning
3	33.8 %	56.1 %
8	47.6 %	66.5 %
12	63.8 %	75.6 %

The above table shows that the reduction of latency grows almost linearly with a coefficient of 3.48 for NG and 2.26 for NG-Pruning, obtained by linear regressions of least square methods.

2) Smaller Neighbor-per-Channel Density Helps: Neighbor-per-channel density is the average number of neighbors per channel. The smaller this value is, the better NG algorithm outperforms the observed-scanning algorithm, shown in Fig.12. But when neighbor count grows above channel count (on the right side of vertical dotted lines), the local channel reuse increases resulting in less performance gain of NG algorithm over observed-scanning.

3) Better Pruning with Increasing Number of Neighbors: The Fig. 13 shows the performance differences between NG and NG-pruning versus neighbor counts. As shown in the graph, pruning algorithm performs better with more number of neighbors. This implies that increasing neighbor counts promotes the ratio of the degree of local non-overlap graph to neighbor counts. This is the case when the number of overlapping edges per each neighbor is limited to a small number, for example, two in our simulations. Let OD denotes the upper bound of this overlapping degree and NB denotes neighbor count. Then, the ratio of non-overlap degree to neighbor count is expressed as,

$$\frac{NB - OD}{NB} = 1 - \frac{OD}{NB}$$

which grows as neighbor count increase. The Fig. 14 shows the proportional relationship of pruning performance and non-overlap degree to neighbor count ratio, similar but extended results of the experiments, shown in Fig. 8

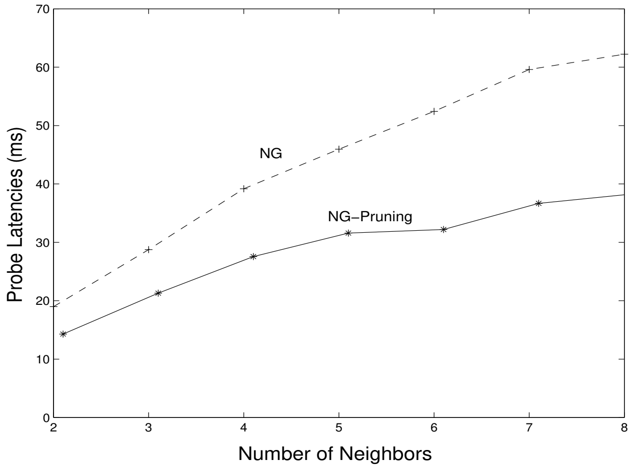

Fig. 13. *The performance of pruning by number of neighbors*

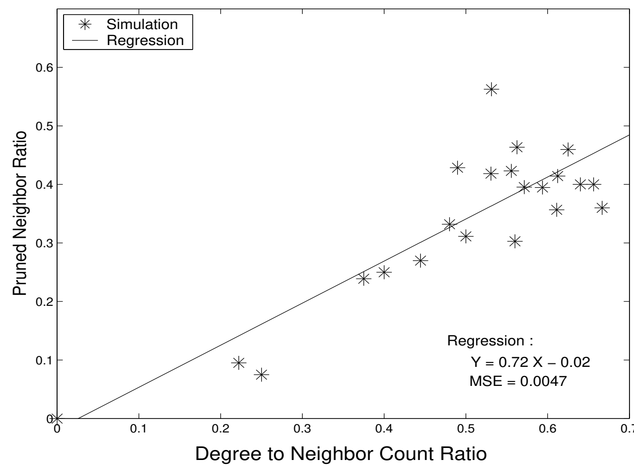

Fig. 14. *Probing Latencies with different Probing Algorithms, and Adjusted Probing Latencies with Chanel-switch-Tx overhead = 10 ms*

VIII. Related Work

The handoffs in WLAN have been consistently reported to be inefficient in the literature [10] [9] [6]. [10] measures the impact of handoff process on layer-3 bandwidth. They observed that a periodical probing on every second reduces the bandwidth of ongoing communication by 15%. Also UDP stream was delayed for 1 seconds while the station handoffs. More precise measurements of Layer-2 handoffs were reported in [9]. By sniffing the medium while the station handoffs, they find the handoff latencies between 53ms to 420ms. Among the components of handoff latency, probing latency is identified to be the dominating factor. Their experiments also show that the handoff latencies vary significantly depending on vendors of wireless cards and access points. [6] reports the consistent results by measuring Layer-2 handoffs from 120ms to 158ms. In addition, they measured Layer-3 (Mobile IPv6 [26]) handoff latencies between 2.9 and 4.7 seconds.

The previous studies have tried to reduce handoff latency in various aspects. Fast handoff schemes in layer-3, especially with Mobile IPv6 have been suggested in [33] and [6]. [33] and [6] exploits relatively fast layer-2 handoff to facilitate layer-3 handoff. In [33], rather than waiting for routing advertisement(RA) or simply increasing the frequency of RA, the scheme allows the layer-2 of the station to initiate router solicitation (RS), explicitly asking the start of layer-3 transition. [33] also employs tunneling between access routers to avoid packet loss.

To reduce layer-2 handoff, [9] provides several fast-handoff strategies such as reducing probe-wait latency to optimal values. Our scheme adopts the suggested values : 7ms for MinChannelTime and 11ms for MaxChannelTime.

NeighborCasting in [34] provides a distributed and dynamic datastructure that maintains the list of candidate foreing agencies (FA) to proactively forward data packets during handoffs. In NeighborCasting mechanism, FAs learns about its neighbor FAs by allowing the mobile nodes to report the identity of old FA to new FA. New FA, then notifies old FA of their neighborhood relationthip. By layer-2 triggered data forwarding, layer-3 handoff latency is reduced to be comparable to laeyr-2 handoff latency.

[13] and [25] transfer or distribute security meterial to access points using neighbor graphs. In [13], the security context contained in current AP is proactively distributed to its neighbor APs through IAPP protocol [27], so that the security context is always one hop ahead of the station's movement. Contrast to [13], [25] maintains neighbor graph in a centralized server. Instead of generating PMK of 802.11i authentication [24] at each handoff with high cost of 800ms, the authentication server proactively distributes PMKs to neighbor APs, reducing the authentication latency down to less than 20ms. For freshness of PMKs, each distributed PMKs are generated by a hashing tree.

IX. Conclusion and Future Work

A. Future Work

NG reflects the aggregated mobility patterns of individual users. However, the mobility pattern of each user can be significantly different because of personal habits, location of residence, or even the social relationships between users. In cellular networks, studies show that an individual user's mobility is routine and predictable with higher accuracy than system-wide approaches [35].

A *personal neighbor graph* (PNG), defined in section III-A, is a neighbor graph that reflects the mobility of a particular user. PNG has many encouraging properties : (i) compact and (ii) high predictable. In the future, we plan to research a prediction based probing scheme using PNG. Also we plan to investigate a hierarchical system with NG and PNG for an efficient, accurate and flexible framework for mobility management in wireless networks.

B. Conclusion

The main contribution of this paper is to introduce two novel and efficient probing algorithms, the NG algorithm and the NG-pruning algorithm, both of which perform better than the two most commonly used probing schemes used by major vendors. We implemented four different algorithms, full-scanning, observed-scanning, NG and NG-pruning, under a deployed 802.11b campus WLAN and measured their probing performances. For more general results, simulations with different configurations were conducted and analyzed. Also we discussed generation, and maintenance of neighbor and non-overlap graphs to show the feasibility of our suggested probing algorithms.

The experiment results show that the NG algorithm reduces probing latencies of full-scanning and observed-scanning algorithms by 80.7% and 30.8%, respectively. Also NG-pruning reduces probing latencies of full-scanning and observed-scanning by 83.9% and 42.1% respectively. By adjustments to the experimental overheads to account for our user space implementation rather than a firmware implementation, we claim that our algorithms can achieve probing latencies of less than $50ms$, which provides a strong foundation for fast hand-offs in WLAN supporting VoIP and other multimedia applications.

From experiments and simulation results, we conclude that the algorithms perform best when there are an abundant number of independent channels available such as in 802.11a.

REFERENCES

[1] IEEE, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Standard 802.11*, 1999.

[2] V. Nee, "New High-Rate Wireless LAN Standards," *IEEE Communications Magazine*, vol. 37, pp. 82–88, Dec. 1999.

[3] D. Corner, J. Lin, and V. Russo, "An Architecture for a Campus-Scale Wireless Mobile Internet," Tech. Rep. CSD-TR 95-058, Purdue University, Computer Science Department.

[4] A. Hills and D. Johnson, "A Wireless Data Network Infrastructure at Carnegie Mellon University," *IEEE Personal Communications*, vol. 3, pp. 56–63, Feb. 1996.

[5] P. Bahl, A. Balachandran, and S. Venkatachary, "Secure Wireless Internet Access in Public Places," in *Proceedings of IEEE Internationa Conference on Communications 2001*, June 2001.

[6] T. Cornall, B. Pentland, and P. Khee, "Improved handover performance in wireless mobile IPv6," in *Communication Systems, 2002. ICCS 2002. The 8th International Conference on*, vol. 2, pp. 857–861, Nov. 2002.

[7] International Telecommunication Union, "General Characteristics of International Telephone Connections and International Telephone Circuits." ITU-TG.114, 1988.

[8] R. Shirdokar, J. Kabara, and P. Krishnamurthy, "A QoS-based indoor wireless data network design for VoIP," in *Vehicular Technology Conference, 2001. VTC 2001 Fall. IEEE VTS 54th*, vol. 4, pp. 2594–2598, Oct. 2001.

[9] A. Mishra, M. Shin, and W. Arbaugh, "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," *ACM Computer Communications Review, to appear*, 2003.

[10] F. K. Al-Bin-Ali, P. Boddupalli, and N. Davies, "An Inter-Access Point Handoff Mechanism for Wireless Network Management: The Sabino System," in *ICNN 2003*, 2003.

[11] Lucent Technologies Inc., "IEEE 802.11 Channel Selection Guidelines," Tech. Rep. WaveLan Technical Bulletin 003/A, Nov. 1998.

[12] Lucent Technologies Inc., "Roaming with WaveLAN/IEEE 802.11," Tech. Rep. WaveLan Technical Bulletin 021/A, Dec. 1998.

[13] A. Mishra, M. Shin, and W. Arbaugh, "Context Caching using Neighbor Graphs for Fast Handoffs in a Wireless Network," Tech. Rep. University of Maryland, Computer Science Technical Report CS-TR-4477. To appear in Infocom 2004, 2003.

[14] T. S. Rappaport, *Wireless Communications*. Prentice Hall PTR, 2002.

[15] C. S. et. al, ed., *3G Wireless Networks*. McGraw-Hill Telecom, 2002.

[16] "Technical specification group services and system aspects; vocabulary for 3gpp specifications (release 6)," Tech. Rep. 3GPP TR 21.905 v6.4.0, Sept. 2003.

[17] G. P. Pollini, "Trends in handover design," *IEEE Communications Magazine*, Mar. 1996.

[18] Mikael Gudmundson, "Analysis of Handover Algorithms," in *IEEE Vehicular Technology Conference, VTC91*, pp. 537–542, 1991.

[19] J. M. H. N. Zhang, "Analysis of handoff algorithms using both absolute and relative measurements," *IEEE Transactions on Vehicular Technology*, vol. 45, pp. 174–179, Feb. 1996.

[20] H. Aida, Y. Tamura, Y. Tobe, and H. Tokuda, "Wireless Packet Scheduling with Signal-to-Noise Ratio Monitoring," in *25th Annual IEEE Conference on Local Computer Networks (LCN'00)*, Nov. 2000.

[21] W. A. Arbaugh, N. Shankar, J. Wang, and K. Zhang, "Your 802.11 network has no clothes," *IEEE Wireless Communications Magazine*, dec 2002.

[22] IEEE, "Standards for local and metropolitan area networks: Standard for port based network access control," *IEEE Draft P802.1X/D11*, March 2001.

[23] B. Aboba and D. Simon, "Ppp eap tls authentication protocol," *RFC 2716*, October 1999.

[24] IEEE, "Draft amendment to standard for telecommunications and information exchange between systems-lan/man specific requirements. part 11: Wireless medium access control and physical layer(phy) specifications: Medium access control (mac) security enhancements.," *IEEE Standard 802.11i*, May 2003.

[25] A. Mishra, M. ho Shin, and W. A. Arbaugh, "Pro-active Key Distribution using Neighbor Graphs," Tech. Rep. Computer Science Technical Report CS-TR-4538, University of Maryland.

[26] D. B. Johnson, C. E. Perkins, and J. Arkko, "Mobility support in IPv6," *Internet Draft draft-ietf-mobileip-ipv6-18.txt, Internet Engineering Task Force (IETF)*, 2002.

[27] IEEE, "Draft 5 Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation," *IEEE Draft 802.11f/D5*, January 2003.

[28] J. Yee and H. Pezeshki-Esfahani, "Understanding wireless lan performance trade-offs," *Communication Systems Design*, pp. 32–35, Nov. 2000.

[29] "Demarc Technologies Group." URL: http://www.demarctech.com.

[30] R. Baird and M. Lynn, "Airjack Driver." http://802.11ninja.net/airjack/.

[31] "High Res POSIX timers." http://sourceforge.net/projects/high-res-timers.

[32] J. Yeo and S. Banergee and A Agrawala, "Measuring traffic on the wireless medium: Experience and pitfalls," Tech. Rep. CS-TR 4421, Dec. 2002.

[33] R. Koodli, "Fast Handovers for Mobile IPv6," *Internet Draft draft-ietf-mobileip-fast-mipv6-08.txt, Internet Engineering Task Force (IETF)*, Oct. 2003.

[34] E. Shim, H. yu Wei, Y. Chang, and R. Gitlin, "Low latency handoff for wireless IP QoS with NeighborCasting," in *Communications, 2002. ICC 2002. IEEE International Conference on*, vol. 5, pp. 3245–3249, Apr. 2002.

[35] L. Perato and K. Al Agha, "Handover prediction: user approach versus cell approach," in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*, pp. 492–496, Sept. 2002.