Abstract — Roaming across heterogeneous wireless networks poses a challenging issue in mobility management such as seamless and efficient handover to reduce end-user’s service interruption. Efficient techniques for seamless handover between access points (APs) when user roams from one area to another can be categorized into three stages in conventional way, i) initiation, ii) preparation, and iii) execution. However, frequent handover may overload the network with signalling traffic and causes call dropping to increase as well. This paper takes a deeper look at predictive triggers at the link layer. With an appropriate implementation of predictive triggering algorithms, early handover initiation and preparation could be performed. This paper addresses the investigation on predictive and accurate mechanisms to generate a handover trigger.

Keywords— Smart Trigger, Predictive trigger, Handover, WiMAX, WiFi, ARIMA

I. INTRODUCTION

In today’s fast-paced, result-oriented world, the trend in communications technology is evolving towards an ubiquitous wireless network environment. Mobile communication devices have evolved into multi-radio device to support of seamless multimedia communication across heterogeneous networks to transfer the end-user’s session from phone calls to data transfer to music and video streaming.

Multiple-radio networking evolution is directed towards full support for seamless roaming between different wireless technologies like 802.11, 802.16, 3GPP, etc. This means that seamless handovers could be easily carried out by mobile devices that support different types of radio networks such as GSM, GPRS, WiFi, cellular, etc. Most of these mechanisms require a smooth and efficient transfer without any obvious data loss and are usually completely transparent to the end-user.

There are two types of handovers: Homogeneous (Horizontal) Handovers and Heterogeneous (Vertical) Handovers. Homogenous or horizontal handovers usually take place within a single access network. Mobility is localized and limited to the network range. Heterogeneous or vertical handovers happen across different network technologies (e.g. cellular to 802.11, 802.11 to 802.16, 3GPP to 802.11, etc.). Hence, mobility is thus global and more opportunistic.

In this conventional handover, Received Signal Strength (RSS) is the major metric used and a good review had been done by Gregory P. Pollhi which is summarized below [1]:

• The base station (BS) with the strongest RSS is selected. In this algorithm, handover may occur although the signal strength of the serving BS is considered good enough. Therefore, it introduces unnecessary handovers.
• Handover occurs when RSS of the serving BS is below a threshold and lower than the RSS of the target BS. By setting a threshold, this algorithm reduces the number of unnecessary handovers.
• Handover happens when RSS of the target BS is greater than the sum of the RSS of the serving BS and the fixed hysteresis margin as well. By adapting a proper hysteresis margin, this approach reduces the ping-pong effects due to the fluctuation of the signal strength that is common for wireless links.
• Handover occurs with the same conditions as above plus the RSS of the serving BS is below a threshold. Such approach combines the advantages of alleviating ping-pong effect and reducing unnecessary handovers.
• A new concept so-called dwelling time is introduced; handover occurs only if the handover condition holds for a certain time.

It is noted that the RSS used in the above algorithms is the averaged RSS. More detailed analyses on the handover performance based on these algorithms were studied in [2]-[4]. Subsequently, Zonoozi M. et al., further optimized the
hysteresis level and signal averaging time [5]. In order to improve handover algorithms, some advanced techniques had been introduced and incorporated into the handover algorithm design recently. These advanced techniques include dynamic programming, neural network, and fuzzy logic [5]-[9].

On the other hand, most of the studies on vertical handover are focused on mobility management, IP layer and above with very few tackling the issues at link layer or below [10]-[11]. Since the technologies associated with WiMAX and WiFi are much different, the existing trigger schemes for horizontal handover that are based on signal strength alone might not be adequate for the case of vertical handover. In view of this, extra metrics such as beacon frame loss and velocity of mobile hosts are considered for the trigger algorithm. Mark Stemm and Randy H. Katz who had considered link layer and below, have proposed handovers to be divided into upward vertical handover and downward vertical handover and specified the differences between the considerations for these two categories as well. They had also proposed the presence or absence of packets to trigger vertical handover, with the primary objective of handover being to minimize handover latency while keeping bandwidth and power overhead low [12]. Similar research had been extended by Qian Zhang et al. which were based on the basic concepts of upward and downward vertical handover [13]. They had introduced physical layer and MAC layer sensing for trigger purpose for the downward handover whereas RSS and adaptive threshold to trigger the handover were employed in upward handover.

In this paper, we focus on the RSS based trigger scheme for handovers. In particular, this paper takes a deeper look at predictive triggers at the link layer. With an appropriate implementation of predictive triggering algorithms, early handover initiation and preparation could be performed and therefore minimize handover delays and service interruptions. Predictive triggers are especially useful for a mobile device in motion where RSS constantly changes. This paper addresses the investigation on predictive mechanisms for handover triggering. The main contribution of this paper is to study the benefits of several methods that are based on Autoregressive Integrated Moving Average (ARIMA); a time series predicting model that based on statistical method, to predict future RSS values and further to predict handover triggers [14]-[16]. Our study is based on real RSS traces collected in office buildings. It is found that by applying ARIMA, we are able to predict future RSS values with high accuracy, especially when we only predict one or two steps ahead. Due to this interest, we expect it may be useful in trigger prediction. Also, we are conducting further research to improve the accuracy of trigger prediction and enable early prediction.

The rest of this paper is organised as follows. Section II provides an overview for the triggering schemes and our proposal of the predictive triggers algorithm. Section III describes time series predicting models i.e., ARIMA. Results discussion is presented in section IV and concluding remark is done in section V.

II. TRIGGER ALGORITHMS AND MECHANISMS

Handovers are usually initiated by a trigger mechanism. Each trigger mechanism has its own unique blend of characteristics. According to Mäkelä and Pentikousis, triggers can be classified into five main criteria, namely: type, origin, occurrence/frequency, event persistence and temporal constraints [17]. There are two main types of triggers that determine the eventuality of whether a handover takes place. The frequency of the trigger events may be either periodic or asynchronous and may be temporary or persistent. However, all these may be based on real-time or fixed constraints that are defined by the algorithm. Table 1 shows some of the event triggers as determined by the IEEE802.21 standard draft [18].

The IEEE802.21 standard draft defines event triggers into three groups: state change events, predictive events and network initiated events. State change and predictive event triggers are smart triggers which work at the data link layer. State change triggers such as Link up (LU), Link Coming Up (LCU), Link Going Down (LGD), Link Down (LD), Link Detected and Link Parameters Change are usually indicators of the current status of the link. On the other hand, predictive trigger such as Link Going Down is the type of trigger which most of us are interested in. This type of trigger could act as a forensic warning of a Link Down.

<table>
<thead>
<tr>
<th>NO</th>
<th>EVENT TYPE</th>
<th>EVENT NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>STATE CHANGE</td>
<td>LINK UP</td>
<td>L2 CONNECTION ESTABLISHED</td>
</tr>
<tr>
<td>2</td>
<td>STATE CHANGE</td>
<td>LINK DOWN</td>
<td>L2 CONNECTION IS BROKEN</td>
</tr>
<tr>
<td>3</td>
<td>PREDICTIVE</td>
<td>LINK GOING DOWN</td>
<td>L2 CONNECTION BREAKDOWN IMMINENT</td>
</tr>
<tr>
<td>4</td>
<td>STATE CHANGE</td>
<td>LINK DETECTED</td>
<td>NEW L2 LINK HAS BEEN FOUND</td>
</tr>
<tr>
<td>5</td>
<td>STATE CHANGE</td>
<td>LINK PARAMETERS CHANGE</td>
<td>CHANGE IN SPECIFIC LINK PARAMETERS HAS CROSSED PRE-SPECIFIED THRESHOLDS (LINK SPEED, QUALITY METRICS)</td>
</tr>
</tbody>
</table>

Predictive triggers, if accurately evaluated:
- could begin the handover process earlier. This would reduce or eliminate service disruption in the continuous stream quality caused by the delay induced by the handover process. This latency is caused by channel scanning delay and reconnecting delay. Channel scanning delay is caused by the mobile station scanning all neighbour channels to collect information about neighbouring APs to identify the next hop AP with the strongest signal. Reconnecting delay is the amount of time it takes the mobile station to reconnect to the new AP [19].
would avoid mis-triggered handovers and thus reduce or avoid the unnecessary cost (payload, time, etc.) involved in scanning for new APs if the current AP signal is still the best among all its neighbouring APs.

Figure 1 shows the handover flow chart with explanation of the functionalities of each block.

In this paper, we set the RSS values for link up threshold \( LU_{TH} = -60 \), link coming up threshold \( LCU_{TH} = -70 \), link going down threshold \( LGD_{TH} = -76 \), and link down threshold \( LD_{TH} = -80 \). The mechanism when the trigger is generated can be summarised as in Table II, where the first column show the previous link status, the first row shows the current smoothed RSS values. The intersection of \( i \)th row and \( j \)th column is the current link status given the previous link status and the current RSS value. Details of the trigger generation algorithm are reported in a separate paper [21].

<table>
<thead>
<tr>
<th>Previous Status</th>
<th>Link Up</th>
<th>Link Up</th>
<th>Link Up</th>
<th>Link Up</th>
<th>Link Going Down</th>
<th>Link Going Down</th>
<th>Link Going Down</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link Up</td>
<td>Link Up</td>
<td>Link Up</td>
<td>Link Up</td>
<td>Link Up</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
</tr>
<tr>
<td>Link Coming UP</td>
<td>Link Up</td>
<td>Link Coming Up</td>
<td>Link Coming Up</td>
<td>Link Coming Up</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
</tr>
<tr>
<td>Link Going DOWN</td>
<td>Link Up</td>
<td>Link Coming Up</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
<td>Link Going Down</td>
</tr>
</tbody>
</table>

A trigger is generated only when the link status changes
- Link status changes to LU \( \rightarrow \) generate a LU trigger
- Link status changes to LCU \( \rightarrow \) generate a LCU trigger
- Link status changes to LGD \( \rightarrow \) generate a LGD trigger
- Link status changes to LD \( \rightarrow \) generate a LD trigger

Raw RSS measurements are captured in various environments such as leaving access point (AP), moving towards AP, moving at the edge, and interleaving of moving and stationary were experimentally collected at Intel Oregon. According to paper [20], the collected raw data can be smoothed by adapting an exponential weighted filter which is expressed as below:

\[ y_t = \alpha y_{t-1} + (1-\alpha)x_t, \]  

where \( y_t \) is the value of the filter at time \( t \), \( x_t \) is the measurement collected at time \( t \), and \( \alpha \) is the smoothing parameter controlling the impact that the current measurements has on the value of the filter; large values of \( \alpha \) will lead to a smooth evolution of \( y_t \), while small values of \( \alpha \) lead to a highly responsive \( y_t \) that reacts to abrupt changes in the behaviour of \( x_t \). We choose \( \alpha = 0.9 \) for smoothing raw RSS [20]. Fig. 2 shows the measurements of RSS when the mobile device was moving away from the AP, then walking around the edge, and walking back towards the AP.

These smoothed RSS can then be used as a baseline to observe when the real triggers occur. All predicted triggers will be compared to the baseline case to test the feasibility of ARIMA.
III. AUTOREGRESSIVE INTEGRATED MOVING AVERAGE (ARIMA)

In IMSL C Numerical Library time series modules, there are three methods which have been developed for the ARIMA models. The theory of the ARIMA can be written as below [14]:

A small, yet comprehensive, class of stationary time-series model consists of the non-seasonal ARMA processes is defined by

\[ \phi(B)(y_t - \mu) = \theta(B)A_t, \quad t \in \mathbb{Z} \quad (2) \]

where \( Z \) denotes the set of integers, \( B \) is the backward shift operator defined by \( B^k y_t = y_{t-k} \), \( \mu \) is the mean of \( Y_t \), and from equation (2), we may obtain

\[ \phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p, \quad p \geq 0 \quad (3) \]

\[ \theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \ldots - \theta_q B^q, \quad q \geq 0 \quad (4) \]

where \( p \) is the number of autoregressive parameters and \( q \) is the number of Moving average parameters. Equations (2)-(4) form a model so-called ARMA(p,q) of order (p,q) [19].

If the “raw” data, \( \{ y_t \} \), are homogenous and non-stationary then they can be differentiated to produce a new set of stationary data, and this model is referred to as ARIMA. Parameter estimation is performed on the stationary time series \( z_t = \nabla^d y_t \), where \( \nabla^d = (1-B)^d \) is the backward difference operator with period 1 and order \( d, d > 0 \). If the data consists of seasonal trend then the advanced and sophisticated model ARIMA \((p, 0, q) \times (0, d, 0)\) is applied, and this model can be written as

\[ \phi(B) \nabla^d_s (y_t - \mu) = \theta(B)A_t, \quad t = 1, 2, \ldots, n. \quad (5) \]

where \( s \) is the seasonal fit parameter, \( \nabla^d_s = (1-B)^d \), and \( B^k y_t = y_{t-k} \). It is noted that \( y_t \) is unobserved, outliers-free time series with mean \( \mu \), and \( \sigma_t \) is the associated white noise. Also, it is assumed that all roots of \( \phi(B) \) and \( \theta(B) \) lie outside the unit circle and when \( s = 1 \), equation (5) reduces to the conventional ARIMA\((p, d, q)\). Hence, the developed function does not treat the series is observable. However, it assumes that the observed values may be contaminated by one or more outliers. From all the assumptions mentioned previously, IMSL has developed three methods with four optional parameters \( p, q, s, \) and \( d \) as follows:

**Method 1 (M1): Automatic ARIMA\((p, 0, 0) \times (0, d, 0)\)**

**Selection**

This method initially searches for the AR\((p)\) representation with minimum Akaike's information criterion (AIC) for the noisy data, where \( p = 0, \ldots, \) maximum number of AR parameters requested. This method ensures every possible combination of values for \( p, s, \) and \( d \) is examined. If \( s = 1 \) and \( d = 0 \), this leads to pure autoregressive prediction.

**Method 2 (M2): Grid Search**

The second automatic method conducts a grid search for a set value of \( p \) and a set value for \( q \). Grid search can be extended to include the candidate values for \( s \) and \( d \). This method does a thorough search for all possible combinations in order to obtain a minimum AIC. However, this would consume lot of time in predicting future data.

**Method 3 (M3): Specified ARIMA \((p, 0, q) \times (0, d, 0)\)**

Model

In the third method, specific values for \( p, q, s, \) and \( d \) are given. If the set values of \( s \) and \( d \) are defined, then a grid search for the optimum values of \( s \) and \( d \) is conducted.

Fig. 3 describes the process of predicting future data \( y_t \) with moving window of size 50 (number of observations). In Fig. 3(a) smoothed data \( y_1, y_2, \ldots, y_{50} \) are fed into the IMSL ARIMA models to obtain future predicted data \( \hat{y}_{51}, \hat{y}_{52}, \ldots, \hat{y}_{560} \). In fact, it is a 10-step ahead prediction; both \( y_6 \) and \( \hat{y}_1 \) are then send to trigger algorithm to observe triggers generation. If triggers generated occur at the same time or within a permitted time range then the proposed smart triggering is meaningful. The predicting process continues by putting smooth data \( y_{11}, y_{12}, \ldots, y_{60} \) into the ARIMA library to
produce the next 10 future predicted data \( \hat{y}_{61}, \hat{y}_{62}, \ldots, \hat{y}_{70} \). This process is clearly illustrated in Fig. 3(b).

IV. RESULTS AND DISCUSSION

Results for RSS values prediction. Figs. 4 illustrate the “macro view” for 1, 5, and 10–step ahead predictions compared to smoothed data in black. All predictions match well with 1, 5, and 10-step ahead predictions. In fact, the accuracy is degrading when the number of step ahead prediction increases. Fig. 5 presents a plot of cumulative percentage vs. absolute error range. Absolute error range is obtained by calculating the absolute difference between predicted RSS and smoothed RRS. It can be seen that error range within (-1,1) is about 98%, error range within (-2,2) is about 99%, and so on and so forth. When number of step ahead prediction increases to 5 or 10, the error range within (-1,1) drops to 76%.

Figs. 6 and Fig. 7 investigate the performance of the three methods. Figs. 6 demonstrates zoom-in graph for the three methods within the time interval (7500, 7800). It is found that the sudden drop/increase of the RSS is hard to predict accurately for all methods. This phenomenon can be clearly seen in Figs. 6 where the particular regions are highlighted.

Among these three methods, M2 outperforms the others. This is evident in Fig.7 where M2 always produces higher cumulative percentage with smaller error range, i.e., (-1,1) implies the RSSs is well predicted. Therefore, missing triggers or unnecessary triggers due to fault prediction can be reduced. From the computational experiments, though M2 outperforms the others, however, it takes the longest time to calculate the predicted RRSs compared to M1 and M3. This is due to the fact that it needs more time to find the best combination of \((p, q, s, d)\) in order to achieve the minimum value of Akaike information criterion (AIC) \([15]\). Interestingly, we find that the smoothed RSSs are very much dependent on \(p\) than the other parameters. Since, most of the time parameters \(q, s, d\) are found to equal 0, 1, and 0 respectively and this forms AR(\(p\)) prediction is then become M3. As a result, it leads to a simple conclusion that AR(\(p\)) is good enough for prediction purpose. It not only brings simplicity but also reduces computational time. As shown in Fig.6, if the trend of smoothed RSSs consists of a sudden drop/increase in magnitude, it is difficult to get an accurate prediction. By looking at the same figure, M1 produces less “up and down” curves compared to M3 (inside the circle of Fig. 6). In terms of accuracy and computational time, M1 is always in between of M3 and M2. From the practicality point of view, it seems that M1 may suit to predict trigger since the time
taken by M1 to complete a prediction is much shorter than M2. Moreover, there is not much difference in cumulative percentage vs error range between M1 and M2 as shown in Fig. 7.

**Results for trigger prediction.** Table III presents the LGD and LD triggers that are obtained in the smoothed RSSs, 1-, 5-, and 10-step ahead predictions (SAPs) which obtained from method M1. The first column represents the time when ‘real triggers’ (it is referred to as baseline) occur that obtained from smoothed RSSs. The second, third, and fourth columns are the time when there are predictive generated triggers for 1-, 5-, and 10-SAPs respectively. The calculation for 5-SAP can be expressed as follows; the window moves 5 steps forward for each prediction, for a window of size 50, smoothed RSSs y1, y2,…,y50 are used to predict ŷ51, ŷ52,…, ŷ55. Then the next prediction use y6, y7,…,y55 to obtain ŷ56, ŷ57,…, ŷ58. This prediction keeps processing till the end. Similar process is applied to the case of 1-SAP and 10-SAPs. In these results, 25000 collected RSSs are tested based on the trigger algorithm mentioned in section II. It is found that there are all together 22 triggers generated; 4 link-up triggers, 6 link-coming-up triggers, 6 link-going down triggers and 6 link-down triggers. It is noted that results for link-up and link-coming-up will not be shown in the following tables since we are more concern about the trigger generation caused by link-down and link-going-down.

Interestingly, for all cases, the number of predicted triggers matches well with the baseline results i.e., 4 link-up triggers, 6 link-coming-up triggers, 6 link-going down triggers and 6 link-down triggers are generated. As can be seen from Table III, the minimum time difference that “real trigger” occurs compared to 1-SAP, 5-SAP and 10-SAP cases is only 100 msec. 1-SAP shows the least time difference compared to baseline results. For the 5-SAP case, results are considered good as well since the maximum time difference is 400 msec. However, for the 10-SAP prediction case we find that the accuracy degrades too much. It implies that when the number of step aheads increases, the accuracy of predicted triggers deteriorates.

V. **Conclusion**

This paper has addressed vertical handovers across multiple wireless networks, in particular, the triggering part of handover procedures. We discussed how to predict signal strength (RSS) and link layer triggers. Prediction of triggers is based on prediction of future RSS values. A few statistical prediction methods have been studied thoroughly by using IMSL ARIMA modules. The number of predicted triggers matches with the baseline results. We found that by applying ARIMA, we are able to predict future RSS values with high accuracy, especially when we only predict one or two steps ahead. We do not expect the occurrence of predictive triggers matches well with the “real triggers” at a particular time but computed results have inspired us to conduct further research in order to improve the accuracy of trigger prediction and enable early prediction. Other issues such as missing and unnecessary triggers due to fault prediction are of interest to study.
### TABLE III: TRIGGER OCCURRENCE TIME

<table>
<thead>
<tr>
<th>Baseline (Real Triggers)</th>
<th>Type of Triggers</th>
<th>1-SAP</th>
<th>5-SAP</th>
<th>10-SAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>4240</td>
<td>LGD</td>
<td>4241</td>
<td>4241</td>
<td>4241</td>
</tr>
<tr>
<td>4282</td>
<td>LD</td>
<td>4284</td>
<td>4286</td>
<td>4301</td>
</tr>
<tr>
<td>5041</td>
<td>LGD</td>
<td>5043</td>
<td>5046</td>
<td>5051</td>
</tr>
<tr>
<td>5110</td>
<td>LD</td>
<td>5111</td>
<td>5111</td>
<td>5111</td>
</tr>
<tr>
<td>8313</td>
<td>LGD</td>
<td>8313</td>
<td>8316</td>
<td>8321</td>
</tr>
<tr>
<td>8359</td>
<td>LD</td>
<td>8361</td>
<td>8361</td>
<td>8361</td>
</tr>
<tr>
<td>12392</td>
<td>LGD</td>
<td>12394</td>
<td>12396</td>
<td>12401</td>
</tr>
<tr>
<td>12415</td>
<td>LD</td>
<td>12416</td>
<td>12416</td>
<td>12421</td>
</tr>
<tr>
<td>13287</td>
<td>LGD</td>
<td>13290</td>
<td>13291</td>
<td>13291</td>
</tr>
<tr>
<td>13365</td>
<td>LD</td>
<td>13366</td>
<td>13366</td>
<td>13371</td>
</tr>
<tr>
<td>13834</td>
<td>LGD</td>
<td>1383</td>
<td>13836</td>
<td>13841</td>
</tr>
<tr>
<td>13858</td>
<td>LD</td>
<td>13860</td>
<td>13861</td>
<td>13861</td>
</tr>
</tbody>
</table>

---

**REFERENCES**


