TRAWL – A Traffic Route Adapted Weighted Learning Algorithm

Enda Fallon[* **], Liam Murphy[*], John Murphy[*], Chi Ma[**]

School of Computer Science and Informatics, University College Dublin, Ireland[*]
Software Research Institute, Athlone Institute of Technology, Athlone, Ireland[**]
E-mail efallon@ait.ie, Liam.Murphy@ucd.ie, j.murphy@ucd.ie, cma@research.ait.ie

Abstract. Media Independent Handover (MIH) is an emerging standard which supports the communication of network-critical events to upper layer mobility protocols. One of the key features of MIH is the event service, which supports predictive network degradation events that are triggered based on link layer metrics. For set route vehicles, the constrained nature of movement enables a degree of network performance prediction. We propose to capture this performance predictability through a Traffic Route Adapted Weighted Learning (TRAWL) algorithm. TRAWL is a feed forward neural network whose output layer is configurable for both homogeneous and heterogeneous networks. TRAWL uses an unsupervised back propagation learning mechanism, which captures predictable network behavior while also considering dynamic performance characteristics. We evaluate the performance of TRAWL using a commercial metropolitan heterogeneous network. We show that TRAWL has significant performance improvements over existing MIH link triggering mechanisms.

Keywords: MIH, vehicular systems, handover, neural networks

1. Introduction

The IEEE 802.21 working group propose the MIH standard [1] to support the communication of network critical events to upper layer mobility protocols. While the MIH standard defines the interface for communication of link layer metrics to upper layer mobility protocols it does not provide specifics on the mechanisms which should be employed to trigger such events. Many existing algorithms [2][3][4] define static thresholds for performance metrics such as RSS. When these thresholds are exceeded events such as Link_Going_Down (LGD) and Link_Down (LD) are triggered. For set route vehicle systems such approaches are limited as they do not consider how the predictable nature of movement enables historic performance metrics to influence predictive link triggering.

In this paper we focus on the optimisation of network handover for set route vehicles such as public transport busses and trains. Such vehicles typically operate in preconfigured routes which are repeated at routine intervals sometimes many times a day. We propose TRAWL, an unsupervised feed forward neural network, which captures repetitive network behaviour while also considering the dynamic performance characteristics of heterogeneous networks.

TRAWL consists of 2 major components; Route Identification and Management (RIM) and an Unsupervised Vehicle Learning Algorithm (UVLA). RIM is responsible for the configuration and removal of routes in the system. RIM also maintains the relationship between Access Points (APs) and vehicle routes. UVLA consists of a feed forward neural network which implements the decision logic for TRAWL. UVLA input consists of a selection of normalised performance metrics. The number of neurons in the output layer is dependent on the network configuration; 1 for homogeneous networks, 2 or more for heterogeneous networks. When the stimulation of a neuron exceeds a user defined activation threshold, the neuron "fires" producing a binary positive output. Positive binary output triggers path switchover to the path specified by the inputs. UVLA aims to maximise throughput per route cycle. In order to determine the learning rate we calculate the rate of change of a linear regression line through historic cycle throughput. A large rate of change results in a large alteration of weights. A small rate of change results in a small alteration of weights.

We evaluate our approach against the standard MIH approach used in [2] using performance metrics from a commercial heterogeneous network installation. The standard MIH approach is limited as uses a static RSS threshold of typically -80dBm to -85dBm to determine when connection termination should occur. For a commercial heterogeneous network implementation where APs are positioned for hot spot coverage, RSS ranging for -80dBm to -90dBm is common. By exploiting historic performance trends TRAWL determines that significant throughput can still be achieved at this RSS level, particularly for uncongested heterogeneous networks. Results illustrate that TRAWL has up to a 400% performance improvement over [2].

This paper is organised as follows; related work is described in section 2. An introduction to neural networks is provided in section 3. In section 4 TRAWL is described. In section 5 and 6 experimental and simulated results are presented. Conclusions and future work is presented in section 7.

2. Related Work

Many existing MIH implementations utilize a performance threshold P_{thres} to generate the MIH LGD event. In such scenarios the relationship between the time that P_{thres} (actual or projected) is exceeded, T_{deg}, and the time at which path handover is initiated, $T_{h\text{-}init,}$ can be expressed as follows:

$$T_{h-init} = \alpha_{\lg d}(T_{\deg})$$

α_{lgd} is an anticipation factor applied to T_{deg} to adjust the aggressiveness of LGD event triggering. Many implementations are based on pre-defined P_{thres}, mostly associated with RSS. If the current RSS crosses P_{thres} the LGD event is generated [2]. The NIST MIH implementation in NS2 [2] utilizes the power level of packets `RXThresh` (P_{thres}) and `Pr_limit` (α_{lgd}) to control event triggering. A number of studies utilize a predictive indication of RSS in this manner [3][4]. In [5] the MN velocity and handover duration are used in conjunction with the predicted RSS level to improve LGD event triggering. While [6][7] use a predictive model which uses the neighbor information to generate timely link triggers so that handover procedures can finish before the link goes down. While these mechanisms utilise metrics which provide a static representation and dynamic view of performance they do not provide a mechanism by which the handover algorithm can tune performance thresholds for changing network conditions.

There are a significant number of recent studies in the area of network handover for vehicle based systems. These studies can be generally categorized in the area of Vehicle Ad-hoc NETworks (VANET) or infrastructure mode network access. Our investigation relates to the latter. In [9], SWiFT focuses on handover optimisation for vehicles travelling greater than 200kmph. SWiFT uses the speed of MN movement and RSS as the basis for link event triggering. It does not consider the condition of the link in the handover decision. [10] proposes a MIP handover mechanism for VANET which maintains the original Care of Address (CoA) configured at the original AP. [11] uses Proxy Mobile IPv6 (PMIP) and Host Identity Protocol (HIP) to reduce handover latency for urban vehicular systems. In [12] we propose multi-homing rather than MIP to pre-configure alternate paths prior to network handover. [13] proposes a collaborative approach in which APs use MN position prediction to limit the potential for retransmission. Such an approach has significant network infrastructure requirements. [14] proposes an architecture for network selection in vehicular systems based on network metrics, user requirements and application QoS. We propose that the predictable nature of public transport vehicles enables performance predictability which is not exploited by any of these approaches. Other studies have investigated how network performance can be optimised by predicting network holes [15][16]. These studies are ad hoc network based and assume an autonomic approach which allows the sensor nodes to self learn/configure. As an end point oriented solution our approach has no ability to change network configuration.

There are a number of ongoing studies which evaluate how artificial intelligence techniques can be used in the optimisation of network handover. [17] [18] propose a mutually connected neural network in order to optimise load balancing and QoS for the entire network. Our work focuses on the optimisation of throughput for client devices. [19] proposes a Hopfield neural network which considers multiple input parameters in the selection of networks. However that study does not consider how the predictable nature of routes used by public vehicles can be used in the optimisation of weights.

3. Artificial Neural Networks

ANN are data processing models which are based on the operation of the brain. The first work on ANN was presented by Mc Cullock and Pitts in 1943 [20]. The work proposed a Threshold Logic Unit (TLU) which used weighted binary inputs. If the weighted sum of inputs exceeded a threshold value, the neuron fired. Many enhancements to the original model have been introduced. The first unsupervised learning approach, Hebbian Learning, was proposed by [21] in 1949. Classification of inputs was introduced by the perceptron model in [22]. The introduction of back propagation enabled the training of synaptic weights based on a desired output. This paper proposes TRAWL, an ANN used to capture historic performance trends for predictable route vehicle systems. These trends are weighted against dynamic metrics.

Fig 1 illustrates a supervised learning ANN. Values x_0, x_1, x_2,.... x_n are provided as input to the neuron. The neuron has 2 modes of operation; training or trained. In trained mode, the neuron applies synaptic weights w_{k0}, w_{k1},.... w_{kn} which enhance or degrade the input values. These weighted values are summed and an activation function φ(.) is applied. φ(.) determines whether the neuron should "fire", producing an output y_k which classifies the input pattern.

Training mode can be implemented through supervised or unsupervised learning. In supervised learning the ANN will have an offline training phase in which neural outputs are compared against a training set. Alterations are made to the synaptic weights to limit the error in classification between the output y_k and the training set d_k. When the ANN correctly classifies the input pattern, the ANN operates in trained mode. Unsupervised learning has no external training patterns. In this mode the ANN self organizes data presented to the network and detects recurrent properties.

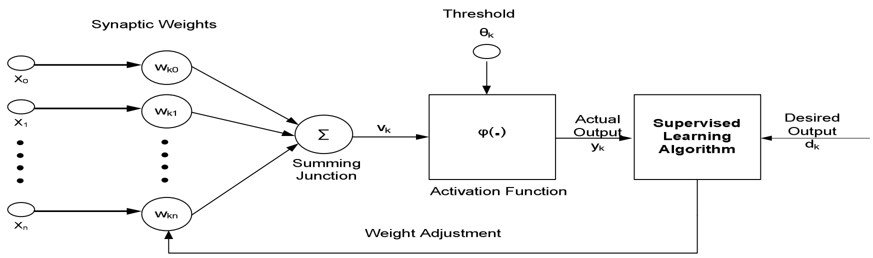

Fig. 1 A Supervised Learning Neural Network

4. TRAWL – A Traffic Route Adapted Weighted Learning Algorithm

Our TRAWL algorithm consists of 2 components:

Route Identification and Management (RIM) – is responsible for the identification and management of vehicle routes. Using the geographical position of the vehicle RIM distinguishes existing, altered or new routes.

Unsupervised Vehicle Learning Algorithm (UVLA) – implements the path selection intelligence within TRAWL. UVLA is a feed forward neural network which operates with a single output neuron for homogeneous networks or 2 output neurons for heterogeneous networks. Back propagation and weight adjustment are implemented each time the vehicle completes a cycle of a route.

The fig 2 illustrates the pseudo code for the TRAWL algorithm. TRAWL dynamically configures and maintains traffic routes using GPS coordinates. Having read the GPS coordinates, TRAWL determines if the current position uniquely identifies a route. If the position is not previously configured, a new route is created and training is initiated. If the position uniquely identifies a previously defined route, the TRAWL algorithm determines if ANN training is required for that route.

TRAWL operates in either training or trained mode. As an unsupervised algorithm TRAWL does not have an offline training phase. TRAWL uses initial end user synaptic weights to determine if handover is required. Following each route cycle the throughput is calculated and synaptic weights are adjusted. TRAWL is trained when (a) the training process does not update synaptic weights (b) synaptic weight updates have no effect on throughput. TRAWL ensures that synaptic weights remain relevant to changing network conditions by applying an `accuracyThreashold`. Throughput is measured for every route cycle and if the `accuracyThreashold` is exceeded training is reinitiated.

```
Struct Route
  param trainingmode = false //training or trained
  param GPS_Cord startOfRoute
  param GPS_Cord[] existingRoute // co-ords for existing route
  param float[] weights  // synaptic weights
  param float activationThreshold //if summation > then fire
  param float[] historicThroughput  //previous cycle throughput
  param float learningRate // rate of altering of synaptic weights
  param float accuracyThreashold // if throughput < reinitiate training

Routine::TRAWL()
  GPS_Cord CurrentPosition = get GPS_Position()
  foreach(Route)  // RIM route management
    if(CurrentPosition contained in Route.StartOfRoute)//cycle complete
      historicThroughput[] += throughputforcurrentcycle
      UVLAcheckAccuracy(historicThroughput)
      if(trainingmode==true)
        UVLATraining()
      else
        UVLAcalculatehandover()
    else if(CurrentPosition contained in Route.ExistingRoute)
      UVLAcalculatehandover()
    else // coords will form a new route
      if(start of new route)
        create Route newRoute
        newRoute.StartofRoute = CurrentPosition
      newRoute.ExistingRoute[] += CurrentPosition
      if(trainingmode==false)
        trainingmode = true
        UVLAcalculatehandover()
      else if(trainingmode==true)
        UVLAcalculatehandover()

Routine::UVLAcalculatehandover()
  foreach(AP)
    param float[] normalisedmetric
    foreach(performancemetric)
      normalisedmetric[] = NormaliseMetric(GetPerformanceMetric())
    param activationValue = (weights[0]*normalisedmetric[0])+......
    if(activationValue>threashold)
      implementhandover(AP)  // neuron fires

Routine:: UVLATraining ()
  param slope = slopeofLinearRegression(HistoricThroughput)
  param errorCorrection = slope*learningRate
  foreach(weight)
    weight+=errorCorrection // alter weights

Routine:: UVLAcheckAccuracy()
  Param slope = slopeofLinearRegression(HistoricThroughput)
  if(abs(slope)>accuracyThreashold)
    trainingmode = true   //reinitiate training
```

Fig. 2 Pseudo code for the TRAWL algorithm

The ULVA model consists of X_0, X_1,... X_n neuron inputs corresponding to the selected performance metrics. Each neuron is a linear threshold gate producing a binary output for path switchover. O_y is defined as follows:

$$V_y = \sum_{i=0}^{N} X_i W_i \quad \textbf{(1)} \qquad O_y = \begin{cases} 1 \ if \ V_y \geq \theta \\ 0 \ if \ V_y < \theta \end{cases} \quad \textbf{(2)}$$

Fig 3 illustrates a UVLA configuration with 2 output neurons for a heterogeneous network with wireless and mobile components.

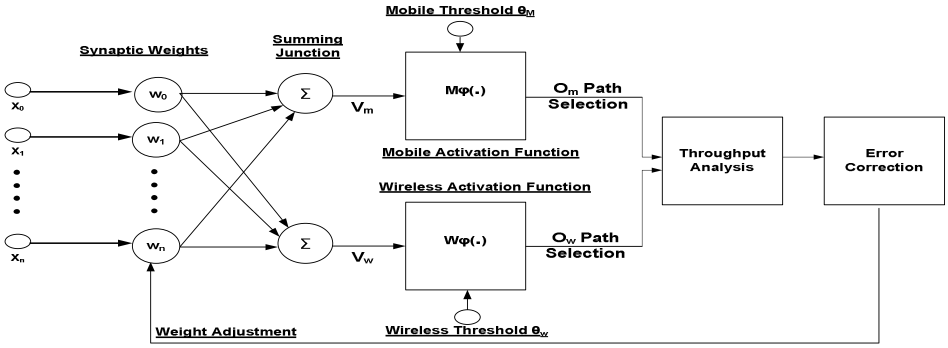

Fig. 3 UVLA Neural Network with 2 Output Neurons

W_i are synaptic weights for each performance metric. V_y is the sum of weighted inputs. θ_y is a user configured activation threshold. If the stimulation of the neuron V_y meets the activation threshold θ_y, the neuron "fires" producing a binary positive output. A positive output indicates that path switchover should occur. The aim is to maximise throughput per route cycle, therefore we calculate the rate of change, c, of a linear regression line for historic throughput as follows:

$$c = \frac{\sum(x-x')(y-y')}{\sum(x-x')^2} \qquad \textbf{(3)}$$

Using c we can determine the rate by which alterations to synaptic weights affect throughput. A positive c indicates that synaptic weight alterations have a beneficial effect on throughput. A negative c indicates that synaptic weight alterations have a detrimental effect on throughput.

In order to control the rate of learning we define a user configurable learning rate constant r. The selection of an appropriate learning rate is critical for the effective operation of the algorithm. If the learning rate is too low the network learns very slowly. If the learning rate is too high weights diverge, resulting in little learning. We define the error correction, ΔW, as the product of c and r.

$$\Delta W = c * r \qquad \textbf{(4)}$$

5. An Experimental Analysis of a Commercial Heterogeneous Network Installation

In recent years heterogeneous networking has gained acceptance as the next logical step in wireless and mobile networking. The ITU have formalized this trend through the fourth generation wireless mobile networks (4G) set of standards [23]. Many mobile operators are embracing heterogeneous networking. Initiatives such as the TeliaSonera Homerun and British Telecom's OpenZone [24] have made heterogeneous networking a reality. In this section we analyze the performance characteristics of one such commercial deployment. Fig 4 illustrates the deployment of APs in Belfast city centre for a tourist bus route. Using NetStumbler [25] we record RSS for all APs as illustrated in fig 5.

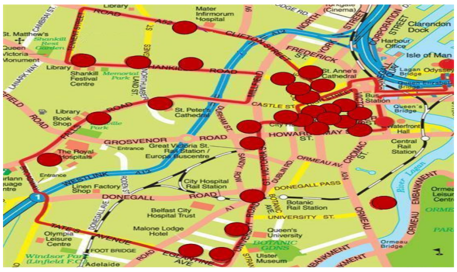

Fig. 4 AP Deployment Belfast City Centre

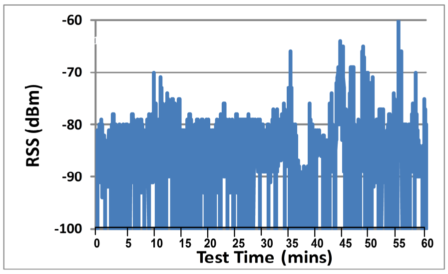

Fig. 5 RSS for APs in Belfast City Centre

Figure 5 illustrates the RSS received by a MN located on the bus as it follows the route outlined in Fig 4 during a 57 minute circular trip. The average RSS over the duration of the test was -85.24dBm. Figs 4 and 5 illustrate a higher concentration of APs in the city centre. In the city centre the average RSS experienced by the MN was -83.23dBm while in the suburbs the average RSS was -90.2dBm. We use the results of this experimental study as input to the simulated model in Section 6.

6. Simulated Evaluation of the TRAWL Algorithm

In this section, we evaluate the performance of TRAWL for the network configuration described in Section 4 using NS2 with the MIH mobility package from NIST [2]. In order to integrate the geographical location of the route in NS2, we record the GPS co-ordinates for all junctions. Using these coordinates we simulate a circular route of 4.82km which is traversed in 57 minutes. The intention is to simulate the download of flat file multimedia content, such as advertising, for display on the next route cycle. We recreate the RSS signature illustrated in Figure 5 in our simulated model. We then evaluate our model in congested and non-congested configurations.

Each AP has a transmit power of 0.281838W, transmit antenna gain of 1, receive antenna gain of 1 and an antenna height of 1.5M. This provides an outdoor signal range of approx 250M. The MIH parameters CSThresh (link detection) and RXThresh (link utilisation) were set to -90dBm and -85dBm respectively. Simulation enhancements as described in [26] were included in the model. The UMTS core network was configured with a 622Mbit link capacity and a delay of 15ms. The WLAN back haul network was configured with a 100MBit capacity and a 1ms delay. The transport layer mobility protocol SCTP was used to implement network mobility. FTP data was then transmitted from the MN towards a back end content server. Fig 6 illustrates the topology of the model.

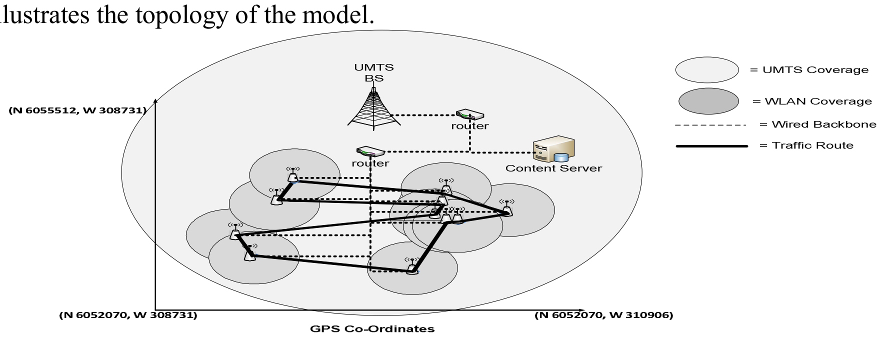

Fig. 6 Simulated Network Configuration

We evaluate the performance of TRAWL for the configuration illustrated in fig 6 with/without congestion for homogeneous WLAN and heterogeneous WLAN/3G network configurations. In all cases we define the initial weights w_l=.35, w_2=.3, w_3=.4, w_4=.25 corresponding to the performance metrics loss rate, RTT, RSS and link bandwidth. These weights reflect our initial understanding of the relative importance of each performance metric. The value range .25 to .4 was chosen since values below this size did not result in the stimulation of neurons and hence no path selection occurred. Figs 7, 9, 11 and 13 illustrate the total throughput and synaptic weights for each of the learning cycles. Figs 8, 10, 12 and 14 illustrate the throughput, rate of learning and error correction for each of the learning cycles. Tables 2-5 are based on figs 7-14 and provide a more detailed view of the configuration of TRAWL parameters for each learning cycle. In order to illustrate the operation of the TRAWL algorithm consider rows 1-3 in Table 2. The initial synaptic weights are set as described above. At routine intervals, at least once per second, the performance metrics loss rate, RTT, RSS and link bandwidth are recorded and normalised. These normalised values are multiplied by the initial synaptic weights. If the summation of these values exceeds the activation threshold switchover occurs. On the first cycle of the route this approach resulted in a throughput of 66.82Mbytes. A slope of a line from the origin to the point (1,66.82) is calculated resulting in a slope of 66.82. This slope is then multiplied by the learning rate, .002, resulting in a weight adjustment of 0.13364. On the second cycle the synaptic weights w_l=.484, w_2=.3, w_3=.4, w_4=.25 result in a throughput of 164.9Mbytes. We calculate the slope of a linear regression line through the points (0,0), (1,66.82), (2,82.45) as 82.45. Multiplying the learning rate .002 by 82.45 results in a synaptic weight adjustment following the second cycle of .1649. Weights are adjusted in this manner following every cycle until the slope of the linear regression line is 0 or until adjustments in the synaptic weights have no effect on throughput.

Local maxima can have a negative effect on ANN performance as learning is centred on a local maximum value. Table 1 illustrates the TRAWL parameters for learning cycles 5-10 for the non congested heterogeneous network. After 9 cycles the slope of the throughput linear regression is 0 indicating that no error correction is required. The synaptic weight configuration w_l=.575, w_{12}=.518, w_3=.590, w_l=.516 has been optimised around a local maximal value. In order to avoid local maxima we introduce a positive or negative random weight adjustment in the range 0-0.2 every 5 learning cycles. In cycle 10 the random weight adjustments w_l=.-.112, w_{12}=-.181, w_3=.-.069, w_l=.-.145 are applied avoiding the local maxima and resulting in the final trained throughput of 74.4Mbytes.

Table 1. Local Maxima in a non congested heterogeneous network configuration

Cycle	w1 (Loss)	w2 (RTT)	w3 (RSS)	w4 (Bw)	Thres	Throughput	Slope	Error Correction	Learning Rate
5	0.582	0.525	0.597	0.523	1	49.15	5.141	0.010	0.002
6	0.592	0.535	0.607	0.533	1	49.15	-3.939	-0.008	0.002
7	0.585	0.528	0.600	0.526	1	49.15	-3.89	-0.008	0.002
8	0.577	0.520	0.592	0.518	1	49.15	-0.928	-0.002	0.002
9	0.575	0.518	0.590	0.516	1	49.15	0	0.000	0.002
10	0.463	0.337	0.521	0.371	1	59.38	2.046	0.004	0.002

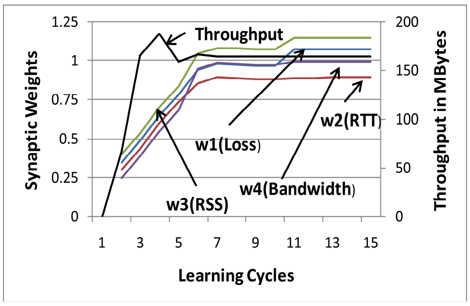

Fig. 7 Throughput and Synaptic Weights for a Non Congested Homogenous Configuration

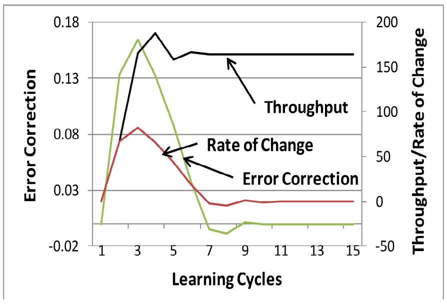

Fig. 8 Throughput and Error Correction for a Non Congested Homogenous Configuration

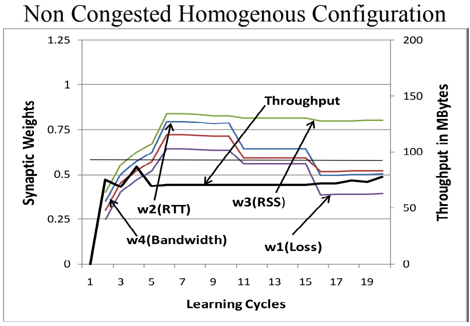

Fig. 9 Throughput and Synaptic Weights for a Non Congested Heterogeneous Configuration

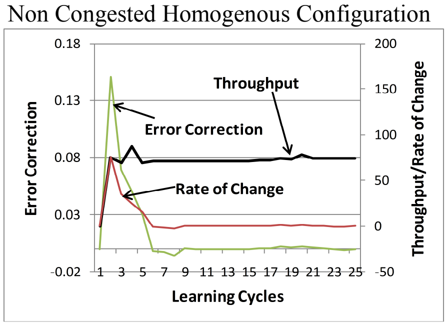

Fig. 10 Throughput and Error Correction for a Non Congested Heterogeneous Configuration

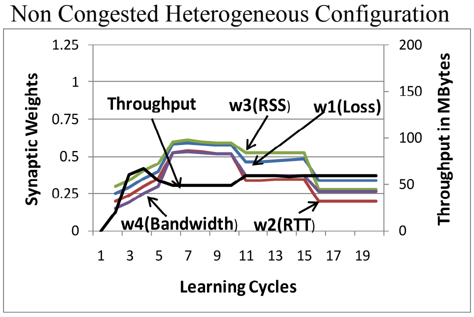

Fig. 11 Throughput and Synaptic Weights for a Congested Heterogeneous Configuration

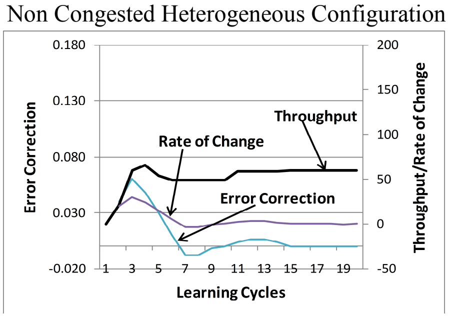

Fig. 12 Throughput and Error Correction for a Congested Heterogeneous Configuration

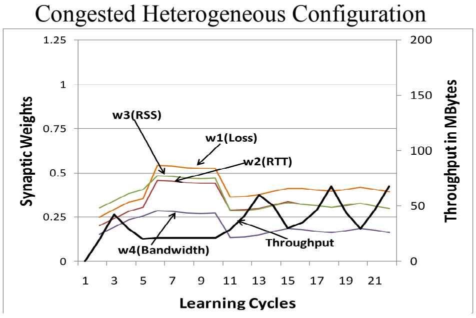

Fig. 13 Throughput and Synaptic Weights for a Congested Homogeneous Configuration

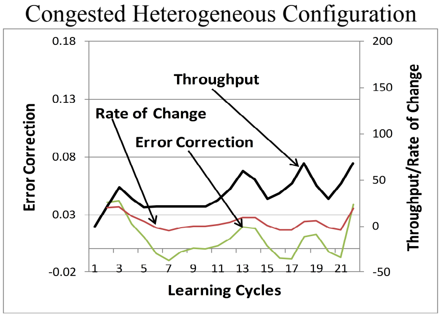

Fig. 14 Throughput and Error Correction for a Congested Homogenous Configuration

Table 2. TRAWL Parameters for a Non Congested Homogenous Configuration

Cycle	w1 (Loss)	w2 (RTT)	w3 (RSS)	w4 (Bw)	Thres	Throughput	Rate of Change	Error Correction	Learning Rate
1	0.350	0.300	0.400	0.250	1	66.82	66.82	0.13364	0.002
2	0.484	0.434	0.534	0.384	1	164.9	82.45	0.1649	0.002
3	0.649	0.599	0.699	0.549	1	188.03	66.217	0.132434	0.002
4	0.781	0.731	0.831	0.681	1	158.49	43.819	0.087638	0.002
5	0.939	0.853	1.045	0.948	1	166.06	19.207	0.038414	0.002
6	0.977	0.891	1.083	0.986	1	164.16	-2.345	-0.00469	0.002
7	0.972	0.886	1.078	0.981	1	164.15	-4.209	-0.008418	0.002
8	0.964	0.878	1.070	0.973	1	164.16	0.943	0.001886	0.002
9	0.966	0.880	1.072	0.975	1	164.16	-0.38	-0.00076	0.002
10	1.071	0.889	1.144	0.996	1	164.18	0.005	0.000010	0.002
11	1.071	0.889	1.144	0.996	1	164.18	0.008	0.000016	0.002
12	1.071	0.891	1.144	0.996	1	164.18	0.006	0.000012	0.002
13	1.071	0.893	1.144	0.996	1	164.18	0.004	0.000008	0.002
14	1.071	0.895	1.144	0.996	1	164.18	0	0.000000	0.002

Table 3. TRAWL Parameters for a Non Congested Heterogeneous Configuration

Cycle	w1 (Loss)	w2 (RTT)	w3 (RSS)	w4 (Bw)	Thres	Throughput	Rate of Change	Error Correction	Learning Rate
1	0.350	0.300	0.400	0.250	1	75.44	75.44	0.150880	0.002
5	0.796	0.722	0.839	0.643	1	70.95	-0.847	-0.001694	0.002
10	0.641	0.591	0.817	0.560	1	70.89	-0.012	-0.000024	0.002
15	0.498	0.517	0.799	0.388	1	72.33	0.288	0.000576	0.002
20	0.568	0.530	0.991	0.528	1	74.4	0.742	0.001484	0.002
25	0.568	0.530	0.991	0.528	1	74.4	0	0.000000	0.002

Table 4. TRAWL Parameters for a Congested Heterogeneous Configuration

Cycle	w1 (Loss)	w2 (RTT)	w3 (RSS)	w4 (Bw)	Thres	Throughput	Slope	Error Correction	Learning Rate
1	0.25	0.2	0.3	0.15	1	20.2	20.2	0.040	0.002
5	0.582	0.525	0.597	0.523	1	49.15	5.141	0.010	0.002
10	0.463	0.337	0.521	0.371	1	59.38	2.046	0.004	0.002
15	0.340	0.200	0.279	0.261	1	59.47	0.063	0.000	0.002
19	0.340	0.200	0.279	0.261	1	59.47	0.063	0.000	0.002

Table 5. TRAWL Parameters for a Congested Homogenous Configuration

Cycle	w1 (Loss)	w2 (RTT)	w3 (RSS)	w4 (Bw)	Thres	Throughput	Slope	Error Correction	Learning Rate
1	0.25	0.2	0.3	0.15	1	20.2	20.2	0.040	0.002
5	0.542	0.457	0.485	0.286	1	21.52	-1.904	-0.004	0.002
10	0.365	0.288	0.284	0.137	1	28.11	1.309	0.003	0.002
15	0.411	0.321	0.322	0.177	1	35.118	-4.1064	-0.008	0.002
20	0.406	0.311	0.310	0.174	1	46.4	-3.829	-0.008	0.002
21	0.394	0.297	0.299	0.162	1	67.89	19.265	0.039	0.002

In Section 2 we described the mechanism used in [2] to implement the MIH LGD event. Table 6 compares the performance of [2] against a fully trained TRAWL.

Table 6. Performance Comparison of NS2 MIH LGD and a Fully Trained TRAWL LGD

	NS2 MIH LGD Throughput (MB)	TRAWL LGD Throughput (MB)	Performance improvement
Non Congested/Homogenous	41.65	164.18	394%
Non Congested/Heterogeneous	71.96	74.4	4%
Congested/Homogenous	42.4	59.47	40%
Congested/Heterogeneous	44.98	67.89	51%

Table 6 illustrates that when TRAWL is fully trained it has significantly better performance than [2] for MIH LGD event triggering, particularly for a non congested homogenous WLAN configuration. In order to explain the significant performance differential we consider fig 5 in more detail. For a commercial heterogeneous network implementation where APs are positioned for hot spot coverage, RSS ranging for -80dBm to -90dBm is common. The average RSS for this installation was -85.24dBm, ranging from -55dBm to -100dBm. -85dBm is considered a point of significant performance degradation in WLAN and `RXThresh` was set accordingly for the standard MIH approach. By exploiting historic performance trends TRAWL determines that the optimal configuration of weights was w_1=1.071(loss rate), w_2=0.895(RTT), w_3=1.144(RSS), w_1=.996(bandwidth). Such a configuration gives the following relative performance metric precedence; loss rate 26%, RTT 22%, RSS 28% and bandwidth 24%. While RSS is considered the most important performance metric, the other metrics have only slightly less precedence. Such a weight configuration determines that significant throughput improvement can be achieved at RSS less than -85dBm. For an unsupervised ANN such as TRAWL, learning is performed online. It is critical therefore that the training of synaptic weights is performed with minimal delay. The optimisation of synaptic weights is a trade off between learning time and throughput. Fig 7, fig 8 and table 2 illustrate that within 2 cycles, the output of TRAWL is within 1% of its trained value. Table 7 compares the performance of a partially trained TRAWL, 2 learning cycles, against [2]. It illustrates that after only 2 learning cycles TRAWL has at least equivalent performance to [2].

Table 7. Comparison of NS2 MIH LGD and Partially Trained TRAWL LGD (2 Cycles)

	NS2 MIH LGD Throughput (MB)	TRAWL LGD Throughput (MB)	Performance improvement
Non Congested/Homogenous	41.65	164.9	396%
Non Congested/Heterogeneous	71.96	69.19	-4%
Congested/Homogenous	42.4	42.09	<1%
Congested/Heterogeneous	44.98	60.28	134%

7. Conclusions and Future Work

In this paper we proposed TRAWL, a feed forward neural network, which uses the predictable nature of public transport routes in order to optimise MIH link triggering. TRAWL uses an unsupervised back propagation learning mechanism which captures predictable network behaviour while also considering dynamic performance characteristics. We evaluate TRAWL using performance metrics from a commercial heterogeneous network. Results presented illustrate up to a 400% performance improvement over traditional MIH link triggering approaches for this network installation. For an unsupervised ANN such as TRAWL the optimisation of synaptic weights is a trade off between learning time and throughput. While a number of cycles are required to fully train TRAWL, we illustrate that early in the training cycle throughput is close to its final trained value. Future work will extend the TRAWL algorithm to consider metrics applicable to media streaming applications.

References

[1] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks – Part 21: Media Independent Handover Services", 21 Jan. 2009.
[2] NIST, "The Network Simulator NS-2 NIST Add-on – Neighbor Discovery", January 2007
[3] V. Mhatre, K. Papagiannaki, "Using Smart Triggers for Improved User Performance in 802.11 Wireless 24 Networks", ACM Mobisys'06, pp.246-259, 2006.
[4] S. Woon, N. Golmie, Y.A. Sekercioglu, "Effective Link Triggers to Improve Handover Performance", IEEE PIMRC06, pp.1-5, 2006
[5] R. Zhang, Y. Hong Wang, K Zhi Huang, H. Quan Wei "Link Quality Oriented Proactive Vertical Handover in Heterogeneous Wireless Networks" CNMT 2009, pp 1 – 4, Jan 2009
[6] S. Yoo, D. Cypher, N. Golmie (2008). "Predictive Handover Mechanism based on Required Time Estimation in Heterogeneous Wireless Networks." MILCOM 2008. IEEE, 1-7.
[7] S. Yoo, D. Cypher, N. Golmie (2008). "Timely Effective Handover Mechanism in Heterogeneous Wireless Networks". Journal, Wireless Personal Communications.
[8] UC Berkeley, LBL, UCS/ISI, Xerox Parc (2005). NS-2 Documentation and Software, Version 2.29.
[9] K.R Kumar, P. Angolkar, D. Das, "SWiFT: A Novel Architecture for Seamless Wireless Internet for Fast Trains", Ramalingam, R. VTC Spring 2008. IEEE, pp. 3011-3015, May 2008
[10] O. Hayoung, K. Chong-kwon "A Robust handover under analysis of unexpected vehicle behaviors in Vehicular Ad-hoc Network" pp. 1-7, Vehicular Technology Conference (VTC 2010-Spring), 2010
[11] S. Céspedes U., X. (Sherman) Shen. "An Efficient Hybrid HIP-PMIPv6 Scheme for Seamless Internet Access in Urban Vehicular Scenarios", IEEE GLOBECOM 2010. Miami, USA.
[12] E. Fallon, Y. Qiao, L. Murphy, G. Muntean, "SOLTA: a service oriented link triggering algorithm for MIH implementations", IWCMC, Caen, France, pp. 1146-1150, 2010
[13] A. Ryo, et.al "Control of transmission timing using information on predicted movement in opportunistic roadside-to-vehicle communication" Proceedings of ICUFN pp. 262-267, 2010
[14] I. Chantaksinopas, P. Oothongsap, A. Prayote "Framework for network selection transparency on vehicular networks" Proceedings of ECTI-CON, pp. 593-597, 2010
[15] N. Ahmed, S. Kanhere, S. Jha "The Holes Problem in Wireless Sensor Networks" ACM SIGMOBILE Mobile Computing and Communications Review Volume 9 Issue 2, April 2005
[16] A. Ghosh "Estimating Coverage Holes and Enhancing Coverage in Mixed Sensor Networks" IEEE International Conference on Local Computer Networks 2004
[17] M. Hasegawa, K. Ishizu, H. Murakami, H. Harada "Experimental evaluation of distributed radio resource optimization algorithm based on the neural networks for Cognitive Wireless Cloud" Proceedings of IEEE PIMRC Workshops, pp. 32-37 2010
[18] M. Hasegawa et.al "Design and Implementation of A Distributed Radio Resource Usage Optimization Algorithm for Heterogeneous Wireless Networks" Proceedings of VTC Fall, pp. 1-7 2009
[19] V. Rakovic, L. Gavrilovska, "Novel RAT selection mechanism based on Hopfield neural networks" Proceedings of ICUMT, 2010
[20] W. McCulloch, W Pitts "A logical calculus of the ideas immanent in nervous activity"
[21] D. Hebb "The Organization of Behavior", published by Wiley
[22] F. Rosenblatt "A comparison of several perceptron models."
[23] "Framework and overall objectives of the future development of IMT-2000 and systems beyond IMT-2000" Recommendation M.1645
[24] S. Heikkilä "Business Models of Mobile Operators for WLAN: Case Examples" available for download www.netlab.tkk.fi/opetus/s383042/2006/papers_pdf/G2.pdf
[25] Netstumbler Network Analysis Software, http://www.netstumbler.com/
[26] C. Palazzi, B. Chin, P. Ray, G. Pau, M. Gerla, M. Roccetti "High Mobility in a Realistic Wireless Environment: a Mobile IP Handoff Model for NS-2" in proceedings of TridentCom 2007.