Mobility Prediction in Wireless Networks using Neural Networks

Joe Capka and Raouf Boutaba
School of Computer Science, University of Waterloo
{jcapka,rboutaba}@uwaterloo.ca
200 University Ave. West, Waterloo, Ontario, N2L 3G1, Canada

Abstract. Wireless network resource use depends in large part on the mobility of network users. The ability to predict this mobility at least in part enables the network to anticipate resource use in the future and take precautionary measures if necessary. This work presents a neural network prediction system that is able to capture some of the patterns exhibited by users moving in a wireless environment and can then predict the future behaviour of these users. These predictions can then be used in a multitude of ways to ensure proper and predictable resource use.

1 Introduction

The popularity of wireless voice communication grew explosively during the end of the last century. The next anticipated step in wireless communication is the delivery of data services, specifically internet services to mobile users. It is anticipated that mobile users in the near future will not only be concerned with the availability of these wireless services, but also with the quality of these services.

One mechanism proposed to aid in providing a certain quality of service is limiting the number of users accessing the network resources at a given point in time. This is known as Admission Control (AC). Many types of AC mechanisms have been proposed for wireless networks [2], [6], [7]. This paper introduces a mobility prediction mechanism that can be used by AC mechanisms to better anticipate the future state of the network and thus make better call accept/reject decisions.

The rest of this paper is organized as follows: Section 2 introduces some background concepts, section 3 provides an overview of current research status, section 4 describes the design of the neural network predictor, section 5 discusses the simulations used to evaluate the predictor performance, section 6 presents the results and section 7 concludes the paper.

2 Background

2.1 Distributed Call Admission Control

Distributed Call Admission Control is a type of AC that uses information from more than the network access point (NAP) the user is currently connected to, and deals with

the granularity of calls. One of the crucial decisions for any dCAC scheme is deciding which of the NAPs will be involved in the admission decision. This decision is not trivial since the group of NAPs that result in the best admission decision can vary according to many factors. These factors can include global data such as the average congestion level in the network, time of day and day of week. It is not difficult to see however that the optimal group of NAPs will be different for each call that is being admitted to the network, and thus more important to determining this optimal group of NAPs is local NAP data as well as specific call data. Local NAP data is data such as traffic patterns observed at a specific NAP, or the congestion of the network in the neighbourhood of the NAP. Data specific to the call being admitted is data such as the resource requirements of the call and the expected call duration.

In trying to devise a way to determine the optimal group of NAPs to involve in a distributed admission decision, it quickly becomes evident that the question being asked is how to best predict which NAPs the MT requesting admission will visit during the lifetime of the call. Therefore the problem has been reduced to one of mobility prediction.

2.2 Mobility Prediction

Intuitively mobility prediction would be the determining of a mobile terminal (MT)'s future location. Although this is the general idea, there are many details that must be specified in order to truly understand what is being described.

The first item that needs further definition is 'location'. The location of the user carrying the MT is primarily thought of as their geographic coordinates. It has been noted however that there are problems with associating the MT's location directly with the user's geographic location. [11], [13] It is better to consider the motion of the MT through the network as the successive list of connections that the MT experiences. What this means exactly is that the MT location is always one of a finite set of locations representing one of the possible access points in the network (NAP). This is illustrated in Figure 1 where the dashed line represents the user mobility and the location of the MT changes as the connection to the network changes from one access point to another. In this particular example, the MT starts at NAP 2, moves through NAPs 5, 6, 7, 11 and ends at NAP 13. It is evident that the idea of a MT's location is greatly simplified by adopting this abstraction from the user's location. The second item worth mentioning is the notion that a prediction is usually based on some previous knowledge. The exact specification of what knowledge is used to make a prediction is very crucial in determining the appropriateness of that prediction scheme. If a prediction is based on data that is simply not available in a given situation, that prediction scheme is useless in that scenario regardless of how well it performs in other scenarios. An example of this is a requirement of privacy. If the prediction scheme has to respect the privacy of users and is only allowed to query their MTs for a very short mobility history, it may not be able to function properly. The data it requires may be present in the system but simply not accessible.

The third item to consider is the classification of the prediction as one that predicts the *time of an event* or one that predicts the *event at a time*. The *time of an event* type

of prediction is one that is presented with an event that is expected to occur and is required to predict the time of this event. An example of this is a prediction mechanism required to predict the time at which a given MT will handoff from one NAP to another. The *event at time* type of prediction is one that is required to predict the state of a system at a given time in the future. An example of this type of prediction scheme is one that is asked what NAP an MT will be connected to at a time t in the future.

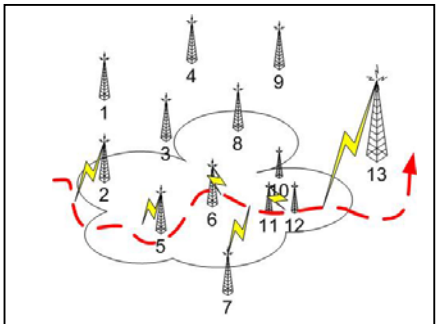

Figure 1

The fourth and last item requiring discussion is the granularity of the prediction. If the prediction mechanism is predicting the time of an event, to what accuracy is the time predicted? Seconds, minutes etc. If the prediction mechanism is on the other hand predicting the event at a time, this event is most likely defined at least in part by a location and thus the granularity of the location needs discussion. This means that a location can be specified in geographical coordinates, a single NAP, a group of NAPs etc.

3 Mobility Prediction Mechanisms

There are many ways of attempting to solve mobility prediction which, result in many prediction mechanisms. Each is unique and developed in order to solve a specific problem or specific type of problem but they are all related and many can be used in scenarios other than those they are proposed for.

There are two main types of wireless networks where mobility is important. These would be a system supported by infrastructure, such as a cellular system supported by base stations, etc. and a system that has no supportive infrastructure, such as ad-hoc networks. The main difference is that an infrastructure supported system can refer to fixed NAP for location while an infrastructure-less system needs an abstract location reference.

Mobility prediction research has mainly focused on supporting the next expected handoff. [1], [3], [10], [12], [15], [16] In reality, MTs will be able to move throughout the network and experience multiple handoffs during the lifetime of a call. It may therefore be necessary to predict more than just the next location the MT visits or the next event the MT will experience. [7],[17],[21]

A large portion of recent research still assumes that user mobility and the connection trace for an MT are strongly dependent. There are a large number of prediction systems that have been proposed which attempt to measure or capture some regularity of the user's mobility in order to extrapolate from this knowledge about the future behaviour of the user's MT [1], [3], [12], [15], [16]. Real life mobility traces have shown that this assumption of user mobility and connection trace of the MT is not as valid as most researchers believe [11], [13]. This raises the issue that it will most likely be necessary to study the behaviour of the MT and its interaction with the network directly.

Another main distinction between prediction systems is whether data is stored on a per user basis or aggregated into some structure. One of the most common per-user types of prediction mechanisms is based on the idea of path recognition [1], [9], [16]. Aljadhai and Znati [17] also use a per user prediction system predicting a user's most likely cluster of locations. Biesterfeld et al. [20] propose a neural net scheme that learns a user's mobility profile to use in prediction. The general argument when using a per-user prediction system is that although the mobility patterns seen in the network as a whole are complex, these patterns become much simpler and more regular when viewed on a per-user basis and can thus be exploited more easily. Prediction schemes that use aggregation argue that the user mobility is subject to geographical constraints at the place of each NAP and thus all users will exhibit similar behaviour at a given NAP. Therefore it is possible to predict the future location of such a user knowing the aggregate behaviour of all or similar users at that NAP. Soh and Kim [5] introduce a prediction technique that uses a road topology database that stores the probability of transfer from one road segment to another.

Another difference in the approaches to solve the problem of mobility prediction seen in current work is whether the prediction produced is based on measurement of user or MT behaviour or matching the pattern of this behaviour with previous behaviour. A measurement based approach will typically compute a probability of events occurring, depending on the value of some parameters. [17] Pattern matching techniques on the other hand attempt to match the observed user behaviour with some previously observed behaviour and forecast the future based on the observed patterns. [1], [9], [16], [17], [18] This distinction is most evident in the per-user type of prediction mechanisms, since most of the schemes that use aggregation will attempt to capture patterns in an overall sense but perform each individual prediction using a measurement of some kind. [10], [12]

4 Predictor Design

We propose a mobility prediction system that uses a neural network to capture connection trace patterns in wireless networks in order to predict future behaviour of MTs. While there has been some research that uses neural networks for similar purposes [19], [20], [21], our approach is novel in that our predictor learns general patterns present at NAPs as opposed to user specific patterns. As proposed in [5], [10], there is good reason to believe that mobility patterns will be influenced in a significant way by the geographic limitations and trends present at the location of a NAP.

We also note that although user mobility may be quite regular, there is a significant amount of independence between this regularity and regularity in the connection trace of an MT belonging to such a user [12], [14]. MT connection traces are influenced by the state of the network and there may be regularity in how a network behaves at particular locations which is a regularity of the network as opposed to a regularity of user mobility. This regularity is impossible to extract from user behaviour. This is best illustrated with a simple example in Figure 2, where a user, represented by the dashed line, initiates a call and connects to either NAP 5 or NAP 2 depending on the state of the network. The handoffs then proceed as illustrated, either from even numbered NAP to even numbered NAP or odd numbered NAP to odd numbered NAP due to some property of the network. Knowing the exact mobility of the user will not determine exactly the behaviour of the MT connection since each time this route is taken, there are two possible connection traces. Studying the MT connection directly, the network regularity present can be observed. As a consequence, our prediction system trains a separate neural network at each NAP using short connection trace histories of MTs that connect to that NAP. The aim is to capture general patterns in local connection behaviour and use these to predict future behaviour of MTs that connect to this NAP. There are a number of advantages to using a generalized pattern recognition mechanism as opposed to a user specific one. First there is no need for each user to build up an individual history since there is an expectation that MT traces for users traveling along similar paths will share localized patterns. Second, a general pattern predictor is better able to handle erratic behaviour by a single user. In a per-user prediction system, a user that behaves in a way he/she never has before will cause the predictor to perform poorly since this type of behaviour is not incorporated into the knowledge the predictor has about that user. In a general pattern predictor, there is at least a chance that some other user has behaved in a way similar to the erratic user and thus the predictor is somewhat prepared for such behaviour. Third, there is an issue of privacy that may arise when keeping track of individual user travel patterns. This issue is not present when only immediate history is used to train a neural network and thus aggregate the individual user travel patterns into more general ones.

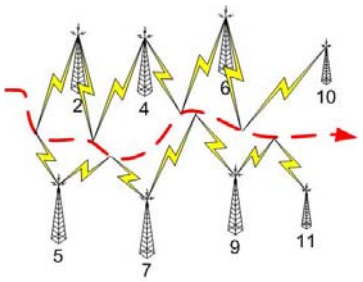

Figure 2

The complete predictor consists of a number of components built around the neural network classifier. Initially it is necessary to collect and convert MT connection traces into data that can be used to train the neural network. The neural net is then trained with this data and ready to predict. The predictions from the neural net are not however of a form directly understandable and have to be translated into a set of

NAPs that the MT is expected to move to in the future. Also, the neural net is trained to predict only the near future, so there is an external mechanism that allows for a recursive prediction farther into the future. Before these components are described in detail, some terminology must be introduced and a few concepts defined.

4.1 Concepts

The predictor views MT connection traces on a discreet time scale of a chosen time unit *t*. This time unit is known as a *time step*. The time step scale needs to be small enough such that MT connection traces are approximately continuous.

A network only experiences handoffs if there is more than one NAP present. NAPs that can handoff to each other are considered direct neighbours. NAPs that cannot handoff to one another are indirect neighbours of a certain degree, this being the smallest number of handoffs that is required for a call to be transferred between these two NAPs. These neighbour degrees enable the definition of an *n-layer neighbour map*. An n-layer neighbour map of a NAP is the set of all NAPs that have a neighbour degree equal to or less than n. A prediction of the future behaviour of an MT is represented as the set of NAPs that the MT will visit in the future time period specified. This set is referred to as the *Future Location Set*.

Table 1. Parameter Definitions Table

$FLS_t(a)$	The Future Location Set at time *t* time units into the future according to the predictor at NAP *a*.
neighbours(a)	The set of NAPs that are direct neighbours of NAP *a*.
T	The threshold such that an output from a neural net predictor corresponding to a certain NAP will only result in this NAP being included in the FLS if this output is higher than or equal to *T*.
λ	The lambda parameter affects the weight given to the absolute length of time spent at a NAP as opposed to the relative time since that NAP has been visited.
$nno(a \rightarrow b)$	The output corresponding to NAP *b*, from the neural network predictor at NAP *a*.
Histlen	The length of the history vector used to classify the MT. This is as far as the system can see in the past on a per user basis.

4.2 Data Gathering and Input Generation

As an MT hands-off from one NAP to another in the network, it creates a history of NAPs it has visited at times in the past. The exact form of this history is an ID vector

of length *Histlen* as defined in Table 1. This history vector contains in it two types of information about the recent mobility behaviour of that MT. The first piece of information is when this MT connected to which NAPs. Intuitively this information should represent a general direction of the MT's motion. The other piece of information contained in this history vector is how long the MT spent at each NAP it connected to. This intuitively carries information about the speed of the MT. A fast MT will spend less time at NAPs than a slow one.

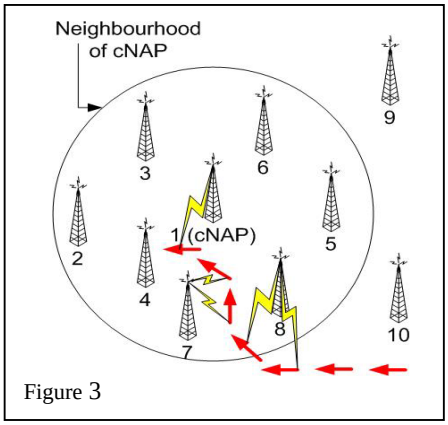

Figure 3

These two pieces of information about the recent connection behaviour of the MT need to be incorporated into one piece of input that can be presented to the neural network. This *input vector* is created as follows: The size of this input vector is defined by the number of neighbours found in the n-layer neighbourhood of the current NAP (cNAP), where the cNAP is the NAP whose neural network is being trained. The parameter n is tuneable and represents how large a neighbourhood the cNAP should be aware of. An n-layer neighbourhood map is created for the cNAP which is a mapping of some network imposed neighbour NAP ID to an index in the input vector. The values of the input vector are then set to an initial value of 0. The parameter lambda (λ) has a value in the range (0, 1], where a value of 1 places all the weight on the absolute time spent at a NAP and disregards any historical sequence and a value close to 0 puts almost all the weight on the historical sequence of NAPs and little on the absolute time spent there. If the present time is $t= 0$, then $t= -k$ is k time steps in the past. The input vector is created by adding the value λ^{k+1} to the value in the input vector that represents the NAP the MT was connected to at time $t= -k$. If this NAP is not in the neighbourhood of the cNAP, the value for that time step is ignored and its information lost. An example can be seen in Figure 3 where each arrow represents one time step in the movement of an MT on its way to the cNAP. The MT is currently in the cNAP for one time step, has spent the previous two time steps connected to NAP 7 and the two before that to NAP 8. The neighbourhood that cNAP is aware of is depicted by a circle, and if the neighbour mapping is such that the ID of the NAP is the position in the input vector and Histlen is 5, then the resulting input vector of this movement is: $[\lambda^1,0,0,0,0,\ \lambda^2+\ \lambda^3,\ \lambda^4+\ \lambda^5]$ (shown transposed to save space) Mathematically the input vector can be expressed as:

$$input = \sum_{k=0}^{Histlen} \lambda^{k+1} \times P_k \quad (1)$$

P_k is a position vector such that it has size equal to the input vector size, and has 0 value for all positions other than the position representing the NAP where the MT was located at time $t = -k$ which is equal to 1. For example, P_1 = [0,0,0,0,0,0,1,0].

4.3 Training Data Generation

In order to train the neural network, it is necessary to have input-output pairs that are examples of what the network is supposed to be able to produce once trained. The input vector of such a training data point is gathered and created as described above. The corresponding output vector of this data point needs to somehow represent the desired prediction output which is in essence some representation of the NAP that the MT will move to in the next time step. This is obviously impossible to obtain at the time the input vector is created since the future behaviour of the MT is not known. This problem is solved by waiting until the next time step to see where the MT actually goes and creating the desired output vector accordingly. This is then coupled together with the already existing input vector and stored as a training data point to be used when training the neural network.

4.4 Output Interpretation

There are essentially two types of output vectors in this prediction system. The first kind are the artificial ones, that is to say the ones created by the system in order to train the neural network. The second kind are the ones that are actually produced by the neural net after it has been trained. These vectors are identical in structure, but there is a difference in the values they hold and in the interpretation of these values.

The size of all output vectors is limited by the 0+1-layer neighbourhood of the cNAP. In other words the size of the output vectors is equal to the number of direct neighbours that the cNAP has, plus one value for itself. This is because the neural net is trained to predict one time step into the future, and there should be no possible way that the MT connects to a NAP other than one of the direct neighbours of the cNAP or itself, by definition of direct neighbour and the scale of time steps.

The artificial output vector is constructed in a manner identical to the P_k vectors described in the input section, except for the stricter size limit. The NAP that the MT moves to in the next time step is what defines which position in the otherwise zero-vector has value 1.

The true output vector as produced by the neural net depends on the characteristic of the neural network, but will have values that range from 0 to 1 in all positions. These are not probabilities even if the numbers would suggest it, although they are similar. These are relative confidence values that the neural net has assigned to the respective NAPs in the direct neighbourhood of the cNAP and to the cNAP itself as potential future locations where the MT will be found in the next time step. From

these, the *Future Location Set* (FLS) is constructed. The FLS is the set of locations, or NAPs, that the neural net has a sufficient confidence in as potential connections of MT in the next time step. Sufficient confidence is defined by a tuneable threshold parameter *T* as defined in Table 1. The higher this threshold, the more critical the predictor is of which NAPs will be part of the FLS. Other than the specifics of the neural network, this is the general one time step prediction mechanism employed in this predictor. In the case of multiple time step prediction, there is an additional step before the comparison of the confidence values and the threshold.

4.5 Neural Network Specifics

At the heart of the predictor lies the neural network. The performance of the prediction system is heavily influenced by the design choices made here. Neural networks are not all alike, and each type of neural net is suited best for a different type of problem. [22]

The type of neural net used in this predictor is a back-propagation network. The main idea behind a back-propagation network is that it starts out with a random pattern encoded in it and as it is trained it modifies this random pattern based on how well the pattern performs on the training data. Depending on how far off the guess is, the network adjusts its internal state and proceeds to the next training point.

There are a few design decisions that need to be made regarding the neural network. First the number of layers in the network needs to be decided. A typical back-propagation network consists of three layers, which is also the number of layers used for this predictor network. There is an input layer, a hidden middle layer and an output layer. Each of these layers server a specific function, refer to [22] for a complete discussion. Then the number of neurons in each layer needs to be decided. The input layer was created such that it could be presented with the input vector, and thus the number of neurons in this layer is the same as the number of entries in the input vector. The number of output layer neurons is similarly dictated by the size of the output vector. The middle layer is then the only layer for which the number of neurons is not dictated by the design of the system.

Second, the transfer functions between the neuron levels need to be decided. There is a number of these, the most common being sigmoid and linear functions. These are mathematical functions that determine how the inputs to an individual neuron determine the output of that neuron. The neural network in this system uses a log-sigmoid function between the input layer and the middle layer and a linear function between the middle layer and the output layer. This configuration is a typical configuration for a back-propagation neural network.

The last major decision that needs to be made is the learning algorithm used to train the network. A typical back-propagation neural network uses a gradient descent algorithm which is what was chosen for the predictor.

The neural network used in this predictor is a typical back-propagation neural net that is quite generic and can likely be improved with studies similar to that in [20]. The focus of this paper is the design of a complete predictor and therefore a generic neural network is sufficient.

4.6 Predicting Multiple Time Steps

So far the system has been described as only able to predict one time step into the future. Since the size of a time step has to be small relative to the call length, the predictor needs to be able to predict farther into the future in order for the prediction to be useful to a dCAC scheme. While it is possible to train a neural network to predict for times farther in the future, such a network is required to capture patterns that are much more complex than those of a one time step prediction and is thus more difficult to create successfully. Due to this, the multiple time prediction system is designed such that only one time step predictions are required. This is achieved using a recurrence relation that defines a prediction of arbitrary time step number as a function of previous predictions. The main idea behind this recurrence relation is that the prediction of some future time step *t* depends on the prediction of the previous time step *t-1*, such that the one step prediction of all the NAPs from *t-1* will create the prediction sought after for time step *t*. The exact recurrence relation is:

$$FLS_t(a) = \{(b, w_b) \mid b \in neighbours(FLS_{t-1}(a)) \wedge w_b > T\} \tag{2}$$

$$w_b = nno(a \rightarrow b) \tag{3}$$

The only information missing in equation 3 is how w_b is computed. There are three methods of combining the outputs from multiple NAPs in order to compute w_b. The combination method presented in equation 4 represents a method based on the idea of voting. Each NAP in the FLS at time *t-1* produces some output confidence value for all its direct neighbours. Then all the output confidence values for a given NAP are added up, and if there is enough combined confidence, the NAP in question is included in the FLS for time *t*. A potential problem with this method is that an error in prediction at one time step that produces a large FLS will propagate and create a very large FLS from that point on.

$$w_b = \left(\sum \left[nno(c \rightarrow b)\right] \forall c \in FLS_{t-1}(a)\right) \tag{4}$$

The combination method presented in equation 5 attempts to prevent the potential problem with the voting method. This is done by taking the maximum individual confidence value as the confidence value compared to the threshold. In essence it means that there is some NAP that expects the MT to travel to the NAP being predicted with a confidence that is enough such that the predicted NAP will be included in the FLS for time t.

$$w_b = \left(Max\left[nno(c \rightarrow b)\right] \forall c \in FLS_{t-1}(a)\right) \tag{5}$$

The last combination method presented in equation 6 is similar to the way probability is computed for multiple events. This method is identical to the voting method except it includes a weight on each vote. This means that the confidence values produced by each NAP are modified according to the confidence with which the NAP was predicted in the previous time step.

$$w_b = \left(\sum \left[nno(c \to b) * w_c\right] \forall c \in FLS_{t-1}(a)\right) \text{ where } w_c \text{ is from } FLS_{t-1} \quad (6)$$

The threshold value to be used in any of these methods has to be determined experimentally with regard to each situation the predictor would be used in.

5 Simulation

In order to validate the performance of the proposed prediction mechanism, a simulation was performed. The simulation consists of a number of parts. The first and underlying part is the mobility model that dictates how users move throughout the network and the structure of the network itself. For this we chose the activity based model as presented in [4]. We modify the network structure however such that there are only 16 NAPs as opposed to the original 45. The network is also constructed such that there are two separate clusters divided by a linear structure. This attempts to model two cities connected by a highway, and is intended to represent the different types of environments that the predictor may have to be used in. The number of users in the simulation is 1000 and each one of these follows the activity based model. The second part of the simulation is the neural network prediction system. Six neurons were used at the middle layer at each NAP, as this number seemed to be reasonable after some initial experimentation. One minute represented one time unit in this simulation. The lambda parameter used to create the input vector was set at 0.5 in order to create a balance between the absolute and relative time spent at every previous NAP. The length of the history vector used was 30 time units, or 30 minutes. The neighbourhood depth each NAP is aware of is two levels. The predictor has been tested on a 7 minute into the future prediction, and as configured, the mobility model allows users to traverse up to three NAPs in the 7 minute interval, thus the two level neighbour maps. The 1000 users were allowed to move around the network for 24 hours during which training data was gathered. After this the neural network at each NAP was trained with the data collected. The next 100 minutes of the simulation were used to test the predictor. Various threshold values were tested at each NAP to discover which would provide the best performance. The range of threshold values tested was [0.15, 0.2]. This range was selected as a result of a number of previous short experiments.

5.1 Evaluation Methodology

Due to the widely varied methods of evaluating mobility prediction mechanisms present in current literature [9], [11], [12], [14] and none of these evaluating the type of prediction that our system produces, we chose to evaluate our system by comparing it to an ideal predictor of the same type as our predictor. What this means with respect to the proposed prediction mechanism is that one and only one NAP should be predicted for any time step. This would be the NAP where the MT will actually be connected to at that time. A predictor that is capable of such accurate prediction is a perfect predictor. We use two numeric parameters to perform this evaluation. The first

parameter is what's called a correctness ratio. This ratio is calculated by comparing the number of times the predicted FLS actually contains the NAP that the MT will be connected to at the time being predicted to the total number of predictions. The second parameter is the predicted set size. The smaller the set size, the more accurate predictor is, as long as the predicted set contains the real future location of the MT. It is obvious that there is a trade-off between the correctness ratio and the set size.

6 Results

Three simulations were performed in total, each one with a different combination method for the inter-time step predictions. As stated each prediction was performed with a number of thresholds. The overall prediction performance per threshold was then calculated using the correctness ratio and set size parameters. The aim was to see how small the average set size could get given a required correctness ratio. Figure 4 shows the smallest average set size per NAP in the network using the different confidence methods for correctness ratio of 0.8. More results are available but not presented due to space constraints. Note that when the correctness ratio could not be reached, it was assumed that the whole network would have to be included in the FLS and therefore the value reflected in the graphs as the FLS size is the size of the network. In order to gauge the complexity of the patterns that are present at the various NAPs, it is important to note that a high number of users were seen in NAPs 5, 7, 8, 9, and 16 while the other NAPs only had a low to moderate number of users pass through them.

6.1 *Result Analysis*

There is a general trend in all the results that the NAPs encountering a large number of unique users are ones where the predictor performs poorly. This can be seen in NAPs 5, 7, 9 10, 16. NAPs 5, 7, 10 and 16 are NAPs that are frequently visited by MTs since they are on the main route in the network (the line from one cluster to the other), but are not in an area that would impose strict geographical restrictions such as NAP 8, 9. These represent the highway like scenario that is somewhat one dimensional and thus imposes considerable geographical limits on motion. We say somewhat represents because MTs can still reach an FLS of size 7 from each of there in the 7 time steps. NAPs 8 and 9 would then be expected to have a predictor with better performance than the other main-line NAPs. The outer cluster NAPs would also be expected to have a well performing predictor since not as many users are expected to pass through them. In general we see that these expectations are met; however there is some unexpected behaviour that happens with each combination method

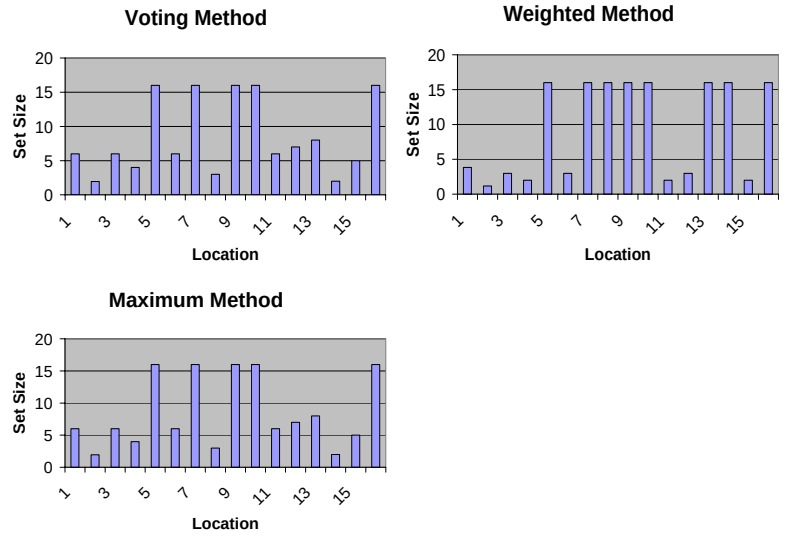

Figure 4

With the Voting method, the outer cluster NAPs have predictors that are able to achieve quite high correctness ratios with FLS sizes that are reasonably small. There are a few NAPs that can achieve over 80% correctness with an FLS size of only 2 or 3, and some of these can achieve even higher correctness ratios with such small FLS sizes, such as NAPs 4 and 14. NAP 8 also has a well performing predictor. The NAPs that are expected to have predictors of a lesser performance quality due to their geographic location also fall within expectation, these being NAPs 5, 7, 10 and 16. NAP 9 was expected to have a predictor comparable in performance to NAP 8, however this is not the case. Further investigation into raw data shows that NAP 9's predictor is of quality comparable to that of NAPs 5, 7, 10 and 16. The reason for this in unclear but there are a number of explanations discussed later on.

The maximum method is identical in performance to the voting method; again most expectations are met as before. Further investigation into the result data shows that this method does vary slightly in some scenarios, but the close similarity to the voting method is intriguing.

The results for the weighted method are quite different than those of the previous two methods, since the predictor seems to work either quite well or not at all. This can be seen when the outer cluster NAPs are considered vs. the NAPs on the main line between the two clusters. Most of the outer cluster NAPs show predictor is with very good performance and FLS sizes of less than four. The main line NAPs and two of the outer cluster NAPs (13, 14) however contain predictors that show very poor performance. This suggests that this method is highly sensitive to the pattern complexity present at a location, since the predictions are either very accurate or not accurate at all.

6.5 General Comments

Is clear that the neural net predictor is very successful in certain situations. It is also clear however that there are situations where the performance is unacceptable. There are a number of potential causes of this. First it is possible that the neural net used is simply too small or too simple to be able to capture the complexity of the patterns present at those locations. It's also possible that there simply are no patterns at those locations or the patterns present are very faint. The possibility is there that the neural predictor simply cannot be used in such situations, however the success of the neural predictor in other situations would lead us to believe that such a conclusion is premature and requires more proof. Although the neural net predictor obviously requires more work in order to be successful in the simulated scenario, we feel it is more important to focus on the development of a mobility model that is more reflective of reality than the one currently used. The extent of mobility models in research today is not sufficient such that a model exists which reflects real mobility in wireless networks [8], [11], [13]. Too many mobility models make assumptions that are not realistic and have an extreme influence on the performance of mechanisms like ours.

7 Conclusion

In this paper we present a mobility predictor that is able to learn and predict connection patterns of MTs and abstracts completely from user mobility. The predictor is an aggregating predictor, in that it does not keep any per user information but rather focuses on using a general behaviour exhibited by MTs at a certain location. This ensures user privacy.

The performance of the predictor was measured using a simulation based on the activity based mobility model [4]. Multiple prediction methods were tested and the general result was that the prediction mechanism is quite successful in some scenarios, while not successful in others. While it is possible to tune the predictor for each simulated scenario, there is no simulation scenario available that is close enough to real mobile networks and thus makes this tuning a marginally useful effort. Therefore the current results are sufficient to conclude that the presented prediction mechanism is useful, and further improvements in its performance will need to be made only after it has been tested either in a real wireless network or with a simulator that is sufficiently reflective of a real wireless network.

References

1. F. Erbas, J. Steuer, D. Eggesieker, K. Kyamakya, K. Jobmann, "A Regular Path Recognition Method and Prediction of User Movements in Wireless Networks", Proc., VTC - IEEE VTS 54th, Fall 2001, Volume 4, pp. 2183 -2187
2. Y. Iraqi, R. Boutaba, "A Novel Distributed Call Admission Control For Wireless Mobile Multimedia Networks", Proc., ACM WoWoMoM, 2000, pp. 21-27

3. X. Shen, J. W. Mark, J. Ye, "User Mobility Profile Prediction: An Adaptive Fuzzy Inference Approach", Wireless Networks 6, 2000, pp. 363-374
4. J. Scourias, T. Kunz, "An Activity-based Mobility Model and Location Management Simulation Framework", Proc., Second ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), 1999, pp. 61-68
5. W.-S. Soh, H.S. Kim, "QoS provisioning in cellular networks based on mobility prediction techniques", IEEE Communications Magazine, Jan 2003, pp. 86- 92
6. J.R. Moorman, J.W. Lockwood, "Wireless call admission control using threshold access sharing", Proc., IEEE GLOBECOM, 2001, Volume: 6, pp. 3698 -3703
7. D. A. Levine, L. F. Akyildz, M. Naghshineh, "A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept", IEEE/ACM Trans. Net, 1997, Vol. 5, No. 1, pp. 1-12
8. T. Kunz, A. A. Siddiqi, J. Scourias, "The peril of Evaluating Location Management Proposals through Simulations", Wireless Networks 7, 2001, pp. 635-643
9. I. R. Chen, N. Verma, "Simulation Study of a Class of Autonomous Host-Centric Mobility Prediction Algorithms for Wireless Cellular and Ad-Hoc Networks", Proc., 36th Annual Simulation Symposium, March 2003, pp. 65 -72
10. K. Curran, G. Parr, "A Framework for the Transmission of Streaming Media to Mobile Devices", Int. J. Network Mgmt, 2002, Vol 12, pp. 41-59
11. J. Chan, A. Seneviratne, "A Practical User Mobility Algorithm for Supporting Adaptive QoS in Wireless Networks", Proc., IEEE International Conference on Networks, Fall 1999, pp. 104 -111
12. A. Jayasuriya, J. Asenstorfer, "Mobility Prediction for Cellular Networks Based on the Observed Traffic Patterns", Proc., 2nd IASTED International Conference Wireless and Optical Communications, 2002, 356-235,
13. J.Chan, B. Landfeldt, A. Seneviratne, P. Sookavatana, "Integrating Mobility Prediction Pre-allocation into a Home-Proxy Based Wireless Internet Framework", Proc., IEEE International Conference on Networks, Sept 2000, pp. 18- 23
14. W. Su, S. J. Lee, M. Gerla, "Mobility Prediction in Wireless Networks", Proc., MILCOM, Oct 2000, Volume 1, pp. 491 -495
15. J. Ye, J. Hou, S.Papavassiliou, "A Comprehensive Resource Management Framework for Next Generation Wireless Networks", IEEE Transactions on Mobile Computing, Fall 2002, Vol 1, No 4, pp. 249 - 264
16. H. Kim, J. Jung, "A Moblity Prediction Handver Algorithm for Effective Channel Assignment in Wireless ATM", Proc., IEEE GLOBECOM, Nov 2001, Volume 6, pp. 3673-3680
17. A. Aljadhai, T. F. Znati, "Predictive Mobility Support for QoS Provisioning in Mobile Wireless Environments", IEEE Journal on Selected Areas in Communications, Oct 2001, Vol 19, No 10, pp. 1915-1930
18. A. Bhattacharya, S. K. Das, "LeZi-Update: An Information-Theoretic Framework for Personal Mobility Tracking in PCS Networks", Wireless Networks 8, 2002, pp. 121-135
19. W.T. Poon, E. Chan, "Traffic Management in Wireless ATM Network Using a Hierarchical Neural-Network Based Predication Algorithm", Proc., 15th International Conference on Computers and their Applications, March 2000,
20. J. Biesterfeld, E. Ennigrou, K Jobmann, "Neural Networks for Location Prediction in Mobile Networks", Proc., International Workshop on Applications of Neural Networks to Telecommunications, 1997
21. B. P. V. Kumar, P.Venkataram, "Prediction-based Location Management using Multilayer Neural Networks", J. Indian Inst. Sci., 2002, 82, pp. 7-21
22. M. T. Hagan, "Neural network design", Boston: PWS Pub., c1996, 1996, ISBN: 534943322