Neuro-fuzzy admission control in mobile communications systems

Raad Raad
University of Wollongong


This paper is posted at Research Online.
7. Neuro-Fuzzy Dynamic Bandwidth Reservation in Micro-Cellular Mobile Networks

7.1. Outline

In the previous chapter, fuzzy logic was used in dynamic bandwidth reservation as well as velocity based admission control. It was shown that fuzzy logic performed well when used for dynamic bandwidth reservation but it was clear that the design methodology of the fuzzy controller needed improvement. The controller was designed using trial and error in the previous chapter. In this chapter, a methodology is presented for designing an adaptive fuzzy logic controller based on neural networks. The Neuro-Fuzzy controller is first trained using data from an approximate analytical model of a micro-cellular network then the controller is fine tuned and adapted to the conditions of a particular cell in the network.
7.2. Introduction

Neuro-fuzzy controllers have previously been shown capable of capturing both high and low level characteristics of complex systems [LIN1996]. This is due to the fact that fuzzy sets allow systems to be described using high-level linguistic terms while back-propagation neural networks can capture the low-level characteristics of such systems (these characteristics of neural networks were discussed in Chapter 3). In this chapter a neuro-fuzzy controller is used to perform adaptive channel reservation in micro-cellular networks where handover rates are expected to be high and non-Poissonian. Further, the neuro-fuzzy controller is able to adapt to the correct number of reserved channels when both the cell dwell time and call holding times have a general distribution. The applicability of this proposed approach is demonstrated for a micro-cellular mobile network with lognormal and gamma distributed call holding and cell dwell times.

With the increasing demand for bandwidth in future Personal Communication Networks, micro- and pico-cellular networks are expected to become prevalent [NAN1993]. Smaller cells make better use of the available frequency through frequency reuse. However, smaller cells lead to an increased number of handovers due to the crossing of many cell boundaries. An increase in the handover rate will result in increasing the probability of dropping a call in progress. In current networks, a small number of fixed guard channels (channels reserved for handover calls) reduce the handover blocking probability at the cost of increasing new call blocking probability [NAN1993]. Reserving a fixed number of channels works well for mobile networks with large cells where the handover rate is small (as seen in Chapters 4 and 5). This is, however, not adequate in mobile networks with smaller cells where the handover rate may vary substantially over time as cells are more influenced by geographic and user
neuro-fuzzy dynamic bandwidth reservation in micro-cellular mobile networks

Mobility factors. Hence in such networks there is a need for adaptive channel reservation to control handover blocking probability.

The major advantage of adaptive channel reservation is its ability to change the number of reserved channels according to the dynamically changing traffic loads. Many variants of adaptive channel reservation have been proposed in the literature. Some schemes make use of channel state information in adjacent cells [Yu1997], others use direction and location prediction to adjust the amount of reserved bandwidth in cells [LEV1997][CHIU2000]. All such schemes suffer from a major disadvantage wherein a large amount of information needs to be transported around the network in order to make reservation decisions. One solution that avoids the problem of excessive data transfer between cells is the use of local information at each based station. This information includes the new call arrival rate, the handover arrival rate and the average channel holding time. Methods that make use of local cell information have been investigated in the literature, e.g. [RAAD2000]. Most available studies dealing with adaptive bandwidth reservation assume that cell dwell times and call holding times are exponentially distributed. While this assumption holds when cell sizes are large, it has been shown both through analytical studies and measurements in mobile networks that this assumption is not accurate for micro-cellular mobile networks. Zonoozi et al [ZAN1997] through the analysis of a mobility model have proposed a generalized gamma distribution for cell dwell times, while Fang et al [YAN1999] have proposed the use of a hyper-Erlang distribution. Traffic studies have also shown that call holding times of current mobile networks follow a lognormal distribution [JED1996]. The handover call arrival rate process is influenced by the cell dwell time distribution. It has been shown that if new call arrivals are Poissonian and cell dwell times are exponentially distributed then the handover arrival process follows a Poissonian distribution [CHL1995]. If cell
dwell times follow a general distribution then this will affect the handover rate process and the assumption that it is Poissonian does not hold any longer. To date very few studies have dealt with adapting the number of reserved channels when call handovers are not Poissonian. In [NAG1996] the scheme is able to approximate non-Poissonian conditions, however, the authors only consider Poisson arrivals and exponential cell dwell times. Non-exponential cell dwell times lead to high complexity in analyzing mobile networks.

Neuro-fuzzy controllers have the ability to deal with such complex problems. A neuro-fuzzy controller is a model free numerical estimator [LIN1996]. Neural networks make the fuzzy system more robust and assuming that representative data is available for training, they can optimize the fuzzy system to correctly match the training set [LIN1996]. When the neuro-fuzzy adaptive approach is applied to mobile networks, each cell can have an independent controller that will allow it to model its unique behaviour based on the mobility and geographical characteristics of the cell. Hence correct quality of service constraints in terms of new call and handover probabilities can be maintained, while allowing better use of the available bandwidth under changing call load conditions.

Fuzzy logic has previously been applied to adaptive channel reservation for tuning the adaptation decision [KO1997] where exponential cell dwell and call holding times were assumed. The approach taken in [KO1997] is to choose the membership functions heuristically without using neuro-fuzzy techniques. Fuzzy logic has also been widely used for handover decision detection (in a signal level detector). In the latter case it is evident that the use of fuzzy logic provides a significant improvement in terms of correct handover detection [ED1995]. Fuzzy logic has also been applied to multi-layer mobile networks for making
decisions about which network layer a user should belong to based on fuzzy descriptions of speed and distance [SHM1999]. Finally, fuzzy logic has been shown to yield significant performance improvements in systems where the dynamics of the system are partially known and when there is unpredictability in the arrival conditions. For example, Bonde and Gizh [BND1996] have shown how a fuzzy controller improves the utilization of an ATM buffer.

The work presented in this chapter allows the adaptation of the correct amount of bandwidth that will guarantee a certain Grade of Service (GoS) in terms of call dropping probabilities for handover calls for mobile networks with non-exponential call holding and cell dwell times. While the fuzzy logic makes ‘soft’ decisions on the amount of bandwidth reserved, the neural net adapts the membership function parameters and the outputs of the fuzzy system. As long as the data set is a realistic representation of the cell behaviour then the neuro-fuzzy controller will work correctly. In the method proposed in this chapter, an initial training table is generated based on the analytical model of the micro-cellular mobile network with the exponential assumptions. The training table is then used to generate a neuro-fuzzy controller. The outputs of the neuro-fuzzy controller are then tuned by an ‘expert’, or through online data collection to account for non-exponential call holding and cell dwell times.

This chapter is organized as follows: Section 7.3 discusses the details of the neuro-fuzzy logic controller as well as an analytical model of the micro-cellular network with bandwidth reservation based on the exponential assumptions for cell dwell times and call holding times. Section 7.3 also describes an algorithm that allows the application of the neuro-fuzzy controller when cell and call holding times are not exponential. Section 7.4 presents simulation results of the
neuro-fuzzy controller for exponential and non-exponential models (lognormal and gamma distributions) while Section 7.5 concludes this chapter.

7.3. Adaptive Channel Reservation Using A Neuro-Fuzzy Controller

7.3.1. Neuro-Fuzzy Controller Structure

The structure of the proposed neural-fuzzy controller (NFC) is shown in Figure 7.1. The NFC takes two inputs, the new call rate $\lambda_n$ and handover call rate $\lambda_h$. The output $y$ is the number of channels to reserve while $z$ is used in the training phase to feed back to the NFC the correct output.
The NFC consists of 5 layers. Layer 1 consists of the input nodes that pass on the inputs to layer 2. Layer 2 nodes perform a fuzzification function [LIN1996]. The new and handover call rates are described by the following linguistic variables (membership functions): Low, Moderate and High input rates. There are six nodes in this layer; each three nodes represent the 3 member functions for each input. The fuzzification function used is a general bell curve and is defined by:

\[ f_j(u_{ij}) = \frac{1}{1 + \left(\frac{u_{ij} - c_{jn}}{a_{jn}}\right)^{2b_{jn}}} \]  \hspace{1cm} (7-1)

where \( u_{ij} \) is the input, \( c_{jn} \) is the center of the generalized bell curve and \( a_{jn} \) and \( b_{jn} \) are the slope parameters. This particular function was chosen because it combines the smooth slope of a bell curve with increased variance (a stretched out bell curve), hence reducing the required number of membership functions (if the bell curve alone was used). Reducing the number of membership functions significantly reduces the number of fuzzy rules resulting in faster convergence in training. It was determined through simulation that 3 membership functions were a good compromise for the NFC. It was found that if only 2 functions are used per input (Low and High), the NFC does not converge to the required output. Using 4 or more membership functions per input resulted in slower training times as well as local minima and maxima in the output surface. It was observed that these (local maxima and minima) are due to incomplete training data (small data set). Using 3 membership functions resulted in an accurate representation of the data set, as well as avoiding the problems that resulted from choosing a larger number of membership functions.
The fuzzified values of Layer 2 are passed up to Layer 3. Layer 3 performs the precondition matching of the fuzzy rules. Each of the rule nodes performs an AND operation. For example, node 1 (the left-most node) of Layer 3 performs the following linguistic operation:

\[
\text{IF } \lambda_1 \text{ is Low AND } \lambda_2 \text{ is Low THEN } u_{11} \text{ is output1.}
\]

output1 is not a fuzzy variable (fuzzy set) and is a real number. Layer 3 of the NFC has 9 IF-THEN rules. These rules are all the possible combinations of the membership functions of the two inputs (new and handover call rates).

Layer 4 has two functions, one during training and another performing the consequence matching of the fuzzy control rules, the OR function. In our proposed NFC, there is only one input to each of the Layer 4 nodes resulting in the output of this layer to be the same as its input. In the training mode, Layer 4 modifies the output values \((\text{Output1},..., \text{Output9})\) according to the reference value passed down through Layer 5. The last layer collects all the outputs from Layer 4 and performs an averaging operation to get a single ‘crisp’ output (real number as opposed to a fuzzy set). Layer 5 has another function during training: to pass the desired output down through the layers, that allows the modification of the membership functions and the outputs.

7.3.2. Micro-Cellular Network Model

A uniform mobile network is considered as shown in Figure 7.2. Mobile terminals have an equal probability of handing over in all directions. All the cells are of equal size and each one has a total capacity of \(N\) channels (units of bandwidth). New calls have access to \((N-h)\) channels, while handover calls have access to all \(N\) channels. If cell dwell times are assumed to be exponential, then the arrival rate of handover calls is Poissonian \([\text{CHL1995}]\). Hence in this case an
analytical model can be used to solve for the new and handover blocking probabilities.

Let \( \rho_H \) be the handover load into each cell given by:

\[
\rho_H = \frac{\lambda_H}{\mu}
\]  
(7-2)

and let the total load be given by:

\[
\rho_T = \frac{\lambda_N + \lambda_H}{\mu}
\]  
(7-3)

where \( 1/\mu \) is the mean channel holding time and it is a combination of the mean cell dwell time as well as the mean call holding time. Using a Markov model, the resulting handover call blocking probability and the new call blocking probability are determined by equations 7-4 and 7-5 respectively:
\[ P_{hb} = \frac{R_{h}^{(N-k)} - R_{T}^{(N-k)}}{N!} \]

\[ P_{nh} = \frac{\rho_{T}^{(N-k)} + \rho_{T}^{(N-k)} \sum_{m=N-h+1}^{N} \frac{\rho_{H}^{m-(N-k)}}{m!}}{G} \]

where

\[ G = \sum_{n=0}^{N-h} \frac{\rho_{T}^{n}}{n!} + \rho_{T}^{(N-h)} \sum_{m=N-h+1}^{N} \frac{\rho_{H}^{m-(N-k)}}{m!} \]

7.3.3. Using NFC with Non-Exponential Cell Dwell and Call Holding Times

As noted earlier, the NFC can model the underlying characteristics of a system. For the case of adaptive channel reservation, it is the number of channels to reserve for a given new and handover call rates that will result in a certain handover blocking probability.

Simulation experiments presented later on in Section 7.4 indicate that an initial training set could be generated based on the exponential model and after several adjustments to the outputs of the NFC, the correct output for non-exponential models can be obtained. New call arrivals are always assumed to be Poissonian.

The first step in obtaining the equivalent exponential model is to find the relationship between the handover rates for the exponential and non-exponential
cases (this is demonstrated in Section 7.4.2 for the case of a lognormal cell dwell time). The NFC is then trained with this initial data set and an initial controller is generated. The outputs (but not the membership functions) of the controller are finally tuned until the correct number of channels that result in obtaining the correct handover blocking probability (which in all our simulations is assumed to be 0.002) are reserved. This process is represented graphically in Figure 7.3 (The detailed process of making the adjustments is presented in Figure 7.9).

7.4. Simulation Results
Simulations were conducted over a ring network (of 20 identical cells) with mobile units having an equal probability of handover in two directions. The mean cell dwell time was chosen to be 30 seconds and the mean call holding time
was chosen to be 100 seconds. These values model a micro-cellular mobile network with a high handover rate. The NFC was initially tested (refer to Section 7.3.1) against an exact exponential model as described in Section 7.2.3. The NFC was then trained to adapt to lognormal cell dwell times and exponential call holding times (Section 7.3.2) and gamma distributed cell dwell times and lognormal call holding times (Section 7.3.3).

### 7.4.1. Testing the NFC against the Exponential Model

The analytical model of Section 7.3.2 was used to generate a training data set that was subsequently used to train the NFC. The trained NFC was then used in a mobile network simulation and the results were compared to those obtained from the exact model. A training table was generated for the NFC using Equation 7.5. The table consists of two inputs and one output. The inputs are the new call rate and the handover call rate, while the output is the number of reserved channels that result in a handover blocking probability equal to or less than 0.002.

Figures 7.4(a) and 7.4(b) show the membership functions of the new call rate before and after training respectively. The corresponding membership functions for the handover rate are shown in Figure 7.5(a) and 7.5(b). The initial membership functions were chosen to cover the desired range of inputs but it is clear that after training, the membership functions have shifted as they converged towards the training data. This is particularly evident in Figure 7.5(b). The output surface of the NFC is shown in Figure 7.6. The figure shows the number of channels that should be reserved for a particular set of inputs. The error in convergence could also be seen, as the surface dips below zero. This was corrected by setting any output below zero as equal to zero. Figure 7.7 shows
that the NFC is able to mimic the analytical model and correctly perform the task of adapting the number of reserved channels that result in the desired grade of service.

Figure 7.4. Fuzzy Membership Functions: “New Call” Rate Before (a) After (b) Training.
Figure 7.5. Fuzzy Membership Functions for “Handover Call” Rate Before (a) and After (b) Training.
Figure 7.6. The Neuro-Fuzzy Output Surface over all the Range of New and Handover Call Rates.

Figure 7.7. Call Blocking Performance for New and Handover Calls.
7.4.2. Applying the NFC to Lognormal Cell Dwell Times

The cell dwell time was modified to reflect a more realistic network situation where the cell dwell time distribution is no longer exponential. The variance of the cell dwell time distribution may depend on the geographical features of the cell as well as on the user mobility distribution. For this case study a lognormal distribution was chosen as an example of a non-exponential distribution. This distribution was applied to cell dwell times so that the resulting call handover process was no longer Poissonian. A mean cell dwell time of 30 seconds and a squared coefficient of variation of 3 were used. The value for the squared coefficient of variation is not critical and values less than 1 can be used as well since the cell dwell time has also been suggested to follow a gamma distribution [CHO1997].

By using the simulation with lognormal cell dwell times the handover rates into the cells were measured and compared to the values obtained when using the exponential model of Section 7.3.2. It was found that a near linear relationship existed between the constant relating the two handover rates and the squared coefficient of variation of lognormal cell dwell times as shown in Figure 7.8.

Hence, by measuring the variance of the cell dwell times, the squared coefficient of variation of cell dwell times can be readily determined and this can be subsequently used to get the equivalent Poissonian rate from the following equation

\[ \lambda_{hPos} = \left( \frac{1}{\alpha} \right) \lambda_{hLog} \]  

(7-7)
where \( \lambda_{hLog} \) is the measured handover rate for the case of lognormal cell dwell times and \( \lambda_{hPos} \) is the equivalent Poissonian handover rate. Having obtained the equivalent Poissonian rate, the analytical model of Section 7.3.2 can then be used to generate a new data set. This data set is then used to generate the new fuzzy model. It is then possible to adjust the output of the resulting model (so that its outputs result in the correct grade of service) in one of two ways,

a) By manually adjusting the outputs of the fuzzy system.

b) By collecting more data online and modifying the training table, the retraining the NFC with the new data set.

Only the manual method is used in this work and method (b) is considered as an extension to the presented work.

Manual adjustments to the outputs in Layer 4 (Output1, …, Output9) are made iteratively until the handover call blocking requirements are met. This is not a long process and it is possible to get an accurate result after the first or second
iteration. This is made possible by the inherent characteristics of fuzzy logic, where upon by making one adjustment, all other outputs in the relative region of the output surface are adjusted in proportion to the membership functions (This is clearly demonstrated later on in Figures 7.13(a) and 7.13(b) in Section 7.4.3). The first step is to modify the most significant output, this is the output (in Layer 4) that is having the most impact on the overall output result (for a particular set of inputs). The second adjustment is made to the next significant output value, and so on until the required output values (that result in the correct grade of service) are reached. One possible way of making such adjustments is shown below in Figure 7.9. It is possible to over-adjust the outputs. This is avoided by checking all the outputs after every adjustment.

![Output Adjustment Process](image-url)

**Figure 7.9 Output Adjustment Process.**
The results of the simulation are depicted in Figure 7-10 and show the blocking performance of NFC-based bandwidth reservation before and after output adjustment. Figure 7.10 also shows that the NFC outperforms fixed bandwidth reservation (FR3, where 3 channels were reserved) as well as Distributed Connection Admission Control (DCAC) proposed in [NAG1996] which uses the number of ongoing calls in adjacent cells to make admission decisions.

Figure 7.11 shows the relative cost of the above four bandwidth reservation schemes. The relative cost combines the two blocking probabilities into a single function and penalizes the dropping of handover calls by considering them to be some multiple of new calls. In the literature this value is usually set to 10 although much higher numbers have also been used [JAB1996]. Hence in our case dropping one handover call is considered equivalent to blocking 10 new calls. Equation 7.8 was used to calculate the relative cost as follows:

$$C = \frac{n_n + 10 n_h}{n_t}$$  \hspace{1cm} (7-8)

where $n_n$, $n_h$ and $n_t$ are the number of new calls, number of handover calls and total number of calls blocked or dropped, respectively.
Figure 7.10. Blocking Performance of Different Reservation Strategies.

Figure 7.11. Cost Function of Various Bandwidth Reservation Schemes.
7.4.3. Applying the NFC to Gamma and Lognormal Cell and Call Holding Times

Another NFC was designed for the case where the call holding time distribution is lognormal (with a squared coefficient of variation, $c^2 = 3$) as suggested by [JED1996] and the cell dwell times were considered to follow a gamma distribution [ZAN1997] (with $c^2 = 0.5$). The processes in Figures 7.3 and 7.9 were followed to design a NFC that results in a handover GoS constraint of 0.002. Figure 7.12 shows the initial results without any adjustments to the NFC outputs, as well as with two successive adjustments, which bring the handover blocking probability closer to the desired GoS constraint. Figures 7.13(a) and 7.13(b) show the initial output surface of the NFC and the output surface after the last adjustment. Figure 7.13(b) shows that less bandwidth needs to be reserved to maintain GoS than first estimated, especially when the loads are high. The adjustments were made to the last three outputs of the NFC (refer to Figure 7.1 and Figure 7.9) and as stated earlier due to the generalizing nature of fuzzy logic, a smooth transition takes place across the output surface.

The above adjustments were made to only 3 outputs of Layer 4 of the NFC. To test the resulting performance of these adjustments at other points of the output surface we chose one of the network cells as a test cell. We set the new call rate in the test cell to 0.9 calls/sec. Subsequently, the new call rates were gradually increased in all the other cells in the mobile network resulting in an increase of handover traffic into the test cell. Figure 7.14 shows that the NFC in the test cell maintained the handover blocking probability below the required GoS. This further highlights the ease with which fuzzy logic allows the adjustment and tuning of reserved bandwidth.
Figure 7.12. New and Handover Blocking Probabilities for NFC.
Figure 7.13. Output Surfaces Before (a) and After (b) the Final Adjustment.
Automated Fine Tuning of the NFC

Earlier it was stated that the output surface of the Neuro Fuzzy Controller could be fine tuned using two methods. One being heuristic, in other words adjusted by an expert who is familiar with the system and the other method being automated or online. Given that cell sizes are getting smaller in order to better exploit frequency diversity resulting in better frequency reuse, generating many cells and manual adjustment of the surfaces may become tedious. The method suggested earlier only requires two or three iterations to arrive at the correct output, but this may become cumbersome when dealing with hundreds and potentially thousands of cells that have interrelated and dependent behavior.

In this section we focus on obtaining the correct amount of bandwidth to reserve given a set of inputs for one given geographical cell to maintain the handover blocking probability at a desired level. It was observed from the
experiments conducted earlier that the output behavior is not consistent at every point of input and produces better results for the handover blocking probability than what is required. For example if we take Figure 12, it can be seen that for new call rates of between 0.8 and 0.9, the handover blocking probability is met at 0.002. But as the new call rate increases, it can be seen that the handover blocking probability outperforms the required value and is well below 0.001. This comes at the cost of an increase in the new call blocking probability. Hence, in the previous section the output functions were adjusted to reduce the amount of reserved bandwidth for values higher than 0.9 calls/sec. This had the desired effect of increasing the handover blocking probability and reducing the new call blocking probability.

Here we propose an automated method of performing this same function. This algorithm mimics the behavior of an expert. For the entire input value set, it compares the output value in terms of the blocking probability to the required value and adjusts the input membership functions relative to the difference in outputs. Hence, if the difference in values is large, then a relatively large adjustment is made and if the difference is small then a small adjustment is made. This process is repeated until the output value, which in this case the handover blocking probability is matched to within a percentage that is specified by the operator of the network. For example a value within 5% could be an acceptable operating point.

As can be seen from previous results (in Figures 7.10 and 7.12), the initial training used an equivalent Poissonian model. The equivalent Poissonian model uses the measured new and handover rates from a general distribution (in earlier cases, the lognormal and gamma distributions were tested) and converts them to a model that assumes Poissonian arrivals and exponential holding times. Using this method, the analytical model could be used to generate a comprehensive data set that will be very difficult to obtain online.
Online data sets are very difficult to obtain as the edge conditions rarely occur and as a service provider, it is hoped that they don’t occur. An edge condition is when blocking occurs, and since this does not occur very often, it is much less represented in a training set. A number of methods in the literature have been suggested to collect online data sets that could be used as a training table [1]. One such method increases the number of edge condition points in the training table once they are found. The method that is proposed here is much more predictable as it is based on analytical model that makes use of measured data. The analytical model for the mobile cells allows the operator to generate a complete data set that covers the entire range of operating values.

\textit{a. The Automated Fine Tuning Algorithm}

**Step 1:** For the entire range of operating points calculate the error between the simulated outputs and desired outputs for both new and handover call blocking probabilities. In this case the output that is being controlled is the handover blocking probability. The desired value so far used in the simulations is for a grade of service of 0.002 for the handover blocking probability. Hence in this step an error value is generated between the desired and actual output of the controller.

A note here about the granularity or number of points that need to be considered from the input space; In the previous heuristic method, it was found that it was sufficient to choose the point where the results deviate the most from the desired value and to simply use this point and adjust the input membership functions around it. This resulted in quite accurate results. Hence, deciding to use many points may not necessarily produce better results. It has been stated earlier because of the interrelationship of the membership functions, adjusting one value results in affecting a reasonably large region of the input space.

\[
\forall \{\ell \in S_n\} \\
E(\ell)_n = O(\ell)_n - D(\ell)_n \\
E(\ell)_h = O(\ell)_h - D(\ell)_h \\
\text{End}
\]
Where $S_n$ is the range of operating values for new and handover calls, $E$ is the error vector generated and $O$ and the $D$ are the actual output and desired outputs respectively. The variable $l$ represents the number of points that are chosen from input space. It would have been desirable to use $n$, but this representation has been reserved for the new calls.

**Step 2:** If both error values show that the actual outputs of the NFC are better than the desired values, then there is no need for any action with regards to this input point.

For $l \{l \in S_n\}$

IF ($E_{nj} \text{ and } E_{nh}$) $\leq 0$ Then

Repeat Step 2 on next point $l$

Else

Go to Step 3

EndIf

Step 3:** In this step, the membership function is adjusted in proportion to the error value obtained in Step 2.

Repeat Until $E \leq \theta$

$f_i(u_{ij},a,b,c) = f_i(u_{ij},a\alpha,b\beta,c)$

$E_b = O_b - D_b$

Generate new $\alpha$ and $\beta$ in proportion to error $E$

EndRepeat

Where $f$ is the generalized bell function for the inputs and $\alpha$ and $\beta$ are scalar multipliers that are proportional to the error $E$, and $u_{ij}$ is the input to layer $i$ in the backpropagation network. The values of $\alpha$ and $\beta$ are adjusted using a smoothing function to ensure convergence. The values for $\alpha$ and $\beta$ were simply taken as a proportion of the error $E$ that was obtained between desired and actual output.
of the Fuzzy Logic admission controller. In this work, it was determined heuristically that $\alpha = \beta = (1 - 0.1E)$. In effect, the values for $a$ and $b$ in Equation 7-9 below were adjusted in small steps in proportion to the error. The value of 0.1 was found to produce a large enough step to obtain the desired results without sending the system into oscillation.

b. The generalized bell Curve

$$f_i(u_{ij}) = \frac{1}{1 + \frac{(u_{ij} - c_{jn})^{2b_{jn}}}{a_{jn}}}$$

There are three tunable parameters in Equation 1, $a$, $b$ and $c$. The $c$ parameter dictates the center of the function. The neural net function of the Neuro-Fuzzy controller adjusts these values in accordance to the training set.

Through the experiments and observations of first method for parameter tuning earlier where the parameters were tuned by an expert in order to produce improved performance, it was observed that tuning the $a$ and $b$ parameters of the functions results in fewer iterations before the output of the controller is improved. The $b$ parameter dictates the ‘width’ of the function for a membership value of 0.5 and the $a$ parameter dictates the width of the function for membership values of 1. Figure 7.15 below shows the above graphically.

![Figure 7.15. The generalized bell curve membership function.](image-url)
By adjusting the values of the $a$ and $b$ parameters, the impact of this membership function on the final output membership functions is either being reduced or increased depending on which way they are varied. In turn, the resultant average from the output membership functions will then adjust the amount of bandwidth to reserve. This was observed quite clearly in the earlier section.

\textit{b. Results}

Figure 7.16 and 7.17 show the resultant membership functions and output surface for the new call rate, handover call and output respectively. This is the result of the original training of the fuzzy logic admission controller based on the poissonian model with adjustments made to the handover rate based on the earlier Equation 7.7. All the settings for the simulation are as in the previous section. After the simulation, the results showed that output was very close to the desired values of 0.002 blocking probability for the handover calls. But it was noticed that at some points the results were either slightly better than 0.002 or slightly higher. For example at a new call rate of 0.82 calls/sec, the handover blocking probability is 0.0017 and that is slightly better than the required value. This comes at the cost of a worse new call blocking probability.

Hence the online adjustment algorithm was set to correct for any values that are outside a tolerance value of (+/-) 0.0002. This value was chosen as a limit, as the granularity of the channel reservation will not allow more fine tuning than this value.
Figure 7.7.26. New Call Rate membership functions.

Figure 7.17. Handover Call Rate membership functions.
Figures 7.19 and 7.20 show the final results of the original fuzzy controller as well as after the running of the adjustment algorithm. Figure 7.19 shows the results with the 95 percentile lines. Each point is the result of 20 different simulations. The first 4 points show lower error tolerance as the simulation was performed over a much longer time frame than the other points in the graph.
Figure 7.20 shows the results without the error lines for clarity.

Figure 7.19. New and Handover blocking probability with error bars.

Figure 7.20. New and Handover blocking probability without error bars.
The algorithm performed two adjustments in one pass through the data set, one that corrected the over reservation for low call rates and one that corrected for under reservation for high call rates. These adjustments are shown below in the form of the membership functions and output surfaces after each adjustment.

**i. First adjustment**

Figures 7.21 and 7.22 show the new call and handover call membership functions after the first pass. Figure 7.23 shows the difference between the handover membership functions between the original training and after the first adjustment. The new call membership functions were left unchanged by the algorithm. The algorithm attempted to adjust the output of the fuzzy logic admission controller in steps of 0.5 channels. As stated earlier, the algorithm works by measuring the error between the desired output and the actual output for the handover blocking probability. The number of points and the granularity of the adjustments are user inputs that allow fine tuning.

For example in the first case where a blocking probability of 0.0017 was obtained instead of 0.0002, the algorithm responded by reducing the output of the fuzzy logic controller by 0.5 channels. In this case there was no need for further adjustments as the output obtained is 0.002. Figure 23 shows that only one of the membership functions was adjusted. The advantages of such a method can be seen as an adjustment of a single point results is a dimensioning adjustment over a larger data set.
Figure 7.21. New Call Rate membership functions after first adjustment.

Figure 7.22. Handover Call Rate membership functions after first adjustment.
Figure 7.23. Difference between Handover Call Rate membership functions between original and first adjustment.

Figure 7.24 and 7.25 show the output surface of the admission controller after the first pass and the difference between it and the original output surface. Figure 25 shows how the output reduction of 0.5 at a single point affected the surrounding output space.

Figure 7.24. Output surface for Fuzzy Logic admission Controller after first adjustment.
Figure 7.25. Difference between output surfaces for Fuzzy Logic admission Controller after first adjustment.

ii. Second adjustment

The second pass corrected for the last input point at a call rate of 1.5 calls/sec. Figures 7.26 and 7.27 show the output surface and the difference in the surface output between the 2nd and first adjustments. Figure 7.28 shows the difference between the original and 2nd adjustment. The figures show that bandwidth was correctly reduced when the admission controller was outperforming the desired result (new call rate equal to 0.8) and the bandwidth was added were the admission controller was not meeting the required criteria (at a new call rate of 1.5).
Figure 7.26. Output surface for Fuzzy Logic admission Controller after second adjustment.

Figure 7.27. Difference between output surfaces for Fuzzy Logic admission Controller after second adjustment.
Figure 7.28. Difference between output surfaces for Fuzzy Logic admission Controller between original and 2nd adjustment.

Figure 7.29. New Call Rate membership functions after second adjustment.
The above algorithm was used to successfully adjust the performance of the fuzzy logic controller by modifying particular points in the output space and relying on the membership functions to smoothly adjust all surrounding points. In this section, an algorithm is investigated where online results are re-entered into the training set to correct for any errors in the fuzzy admission controller.

Hence, the method that will be presented in this section assumes that one can collect enough ‘real’ points and add them to the training table to retrain the fuzzy logic admission controller. In fact such an assumption is not realistic in a real system, where the network operator dimensions the bandwidth in order to avoid the edge conditions or in other words where blocking of new and handover calls is most likely to occur. It is much more likely that a few instances of the edge conditions will be available to the network operator.

In this algorithm, it is assumed that only very few points are available to the network operator to retrain the fuzzy logic admission controller. These few edge conditions are extrapolated by the algorithm in order to generate more data points. We refer to these points as pseudo measurements. This is done to
highlight the fact that they are not real measurements, but approximations extrapolated from very few real measurements. The main assumption made by this algorithm is that the function that it is trying to adapt to is monotonic for small distances around these measured points. In the case of the blocking probability that is clearly the case.

The algorithm works by adjusting the training set of the particular point of concern as well as the surrounding points. If only the single point is actually used it will be drowned out by all other points. If the same point is replicated many times in the training set, it may result in simply causing that particular section of the admission controller to retrain and will not affect other areas which may be in error. This is more evident if many membership functions are used to represent the data set.

d. The algorithm

Repeat steps 1 and 2 until all points are within the acceptable value for the handover blocking probability.

Step 1: For the entire range of operating points calculate the error between the simulated outputs and desired outputs for both new and handover call blocking probabilities. In this case the output that is being controlled is the handover blocking probability. (This is the same step as the previous algorithm).

Step 2: For the input range as specified around the measured point adjust the original training set by applying the retraining function.

For example the training range around a specific measured point could be +/- 5% for all inputs. Hence all points within this range will be modified by the retraining function. The retraining function specifies the magnitude of how the change will occur. For example three possible training functions that could be applied are flat, descending gradient or bell shape. In the case of a flat function, the specific points that were chosen for retraining will simply apply an equal
adjustment to the entire set of points in the range of values. For example if the original training set specified that 10 channels should be reserved for a specific set of inputs, but the measured data indicated that 8 is a better fit, then all points within the retraining range will have their values shifted down by 2.

If the gradient method is chosen, then the points of the training set will be adjusted in proportion to how far they lay away from the measure point. The closer the training data point is to the measured one the more adjustment that takes place. The further away it is, the less adjustment takes place. This is the same for a bell type function that will slowly decay away from the measured point.

e. Results for the second algorithm

Figure 7.31 below shows the new and handover call blocking probabilities for the new retraining algorithm as well as the original and the membership function changing algorithm presented in the previous section. The figure shows that like the previous algorithm, the measured or ‘real’ points (in this case 3) that had been outside the allowable range had been brought back in within the tolerance limit for the handover blocking probability. Hence, the fuzzy logic admission controller has been retrained to reflect the measured data, but in this case only relying on 3 measured points from the ‘real’ system (which is the simulated system in this case). The flat and the simplest retraining functions were used in this case. Only one pass was made through the data as the adjustments that needed to be made are small, and the granularity of the single channel does not allow further refinement.
Figure 7.31. New and Handover blocking probability without error bars for the original admission controller and the two adjustment algorithms.

Figures 7.32 and 7.33 show the difference between the original and retrained surfaces of the fuzzy logic admission controller. Figure 7.33 is the same Figure as 7.32 but from a different viewing point. It is clear that the surface has been adjusted appropriately to reflect the new data. What is interesting though, is that this surface is much smoother than the surface produced by the previous algorithm which made the adjustments to the membership functions rather than the training data. It seems that adjustments to the training data at least in this case maybe more beneficial as this results in a smooth adjustment of all the admission decisions.
Figure 7.32. Difference between output surfaces for Fuzzy Logic admission Controller between original and 2nd algorithm.

Figure 7.33. Difference between output surfaces for Fuzzy Logic admission Controller between original and 2nd algorithm (another view).
The conclusion can be drawn that one can successfully correct for the original training performed using the approximate poissonian model when actual results differ from the desired outputs. This difference is due to the fact the actual system has difference characteristics in terms of arrival rates, call holding time and sell dwell times. These simulation results show that it is quite possible to readily automate the adjustment algorithm to correct for any errors and two possible implementations of the algorithm show some promising results.

![Figure 7.34. The relative cost of the two adjustment algorithms.](image)

Figure 7.34 compares the relative cost of both adjustment algorithms with respect to the non-adjusted case.

Initially the cost for the non-adjusted case, the first and the second algorithms were determined by using the following equation as in previous chapters:

\[ C = P_w + 10P_s \]
The handover blocking probability was considered to cost 10 times more than the new call blocking probability. After all three costs were determined, the cost of the non-adjusted case was used as a base for comparing the two algorithms. This was carried out by subtracting the base line cost from the costs obtained from the two algorithms respectively. The results are shown in Figure 7.34. A negative value on the graph indicates that the algorithms were less costly than the base line un-adjusted performance of the original Neuro-fuzzy admission controller. Conversely a positive value indicates that the cost is higher for the algorithms. From the figure, it seems that the second algorithm results in better performance as the cost seems to be consistently lower than the first algorithm.

7.6. Conclusion

A Neuro-Fuzzy controller was developed to carry out adaptive channel reservation in micro-cellular networks with general cell dwell times and call holding times. To date the analysis of micro-cellular networks reported in the literature has been restricted to the use of exponential call holding and cell dwell times for tractability reasons. The results presented in this chapter showed that the NFC outperforms both DCAC and fixed bandwidth reservation. Two algorithms for designing and training the NFC were also presented. These algorithms were used and tested through simulation in two case studies; One dealing with lognormal cell dwell times and the second with lognormal call holding and gamma cell dwell times. The results showed that the NFC was able to meet the required GoS constraints after one or two adjustments, making the proposed method a practical solution to the problem of improving bandwidth utilization whilst reducing the possibility of a degraded user experience through call drop-outs in mobile networks with a small cell size.
Another major contribution of this chapter is the presentation of two unique algorithms for automatically training the Neuro Fuzzy Controller without the need of human intervention. Both algorithms were shown to perform the job of fine-tuning the data that was retained through the theoretical model by using online data through simulation.