CHAPTER III
METHODOLOGY

Based on the previous chapters, a proposed solution for implementation is formed. The adopted ideas and concepts are Periodic Algorithm, traffic prediction and bandwidth re-allocation. This chapter covers the methodology used to develop a solution for dynamic bandwidth allocation mentioned in the previous chapter.

3.1 Problem Statement and Analysis

There are many solutions for dynamic bandwidth allocation. It can be using algorithms [52-56][59] (allocation methods), mathematic formulas [57] (queuing fairness), statistical methods [60-61], Neural Networks [63-67] or Fuzzy Logic (refer Chapter II). Unfortunately, every solution has its weaknesses. However, these weaknesses can be offset by integrating with another solutions (e.g. the relation between the Neural Network and the Fuzzy Logic).

This research is attempting to incorporate some of the solutions that have been developed previously, and then form a new solution model to the dynamic bandwidth allocation. This solution is based on the integration of a few ideas. These ideas are:

- Periodic Algorithm [59]
    - Due to the overhead of setting up or tearing down of the VCs, it is not practical to change connections' bandwidth by every 0.01 seconds or as the bandwidth-required changes. The Periodic Algorithm [59] is implemented in this case. This algorithm implements the dynamic bandwidth allocation and adjusts the bandwidth periodically (in fixed equal size time interval).
- Neural Networks
    - The prediction of future sample values of a time series can be done by extracting knowledge from its past values [35-41]. Predicting the future in real-time (updating prediction on the fly) has potential applications in such fields as adaptive control and system modelling. In fact, the problem of predicting the

future is extremely general. A Neural Network is constructed to predict the next incoming traffic. This concept is developed by Demuth and Beale [43].

- Fuzzy Logic
  - Fuzzy Logic is integrated to calculate the amount of bandwidth to be re-allocated. This Fuzzy Logic does a simple job by considering the given parameters, which includes next coming traffic (predicted traffic), available bandwidth, and available buffer capacity.

In order to implement dynamic bandwidth allocation in ATM network using Neural-Fuzzy, the following steps must be carried out:

1. Construct a Neural Network for next incoming traffic prediction.
2. Generate training data, which is similar to real traffic. (Of course, traffic captured from real network operation is better for Neural Network training)
3. Train the constructed Neural Network with the training data.
4. Construct a Fuzzy Logic for decision-making (in bandwidth re-allocation).

3.2 Network Traffic Generation

Due to unavailability of real traffic (data), a series of faked network traffic must be generated. This traffic must be similar to the real network operation, because this traffic will be fed into the Neural Network for training. As proposed in [53][67][80-82],

- the holding time for a connection is exponentially distributed
- the ATM packets is in Poisson arrival in a connection when the connection is active.

Exponential Distribution

The exponential distribution with parameter λ ($\lambda > 0$) is given by

$$F(x) = 1 - e^{-\lambda x} \qquad x \geq 0$$

The exponential distribution has the property that its mean is equal to its standard deviation:

$$E[X] = \sigma_X = \frac{1}{\lambda}$$

When this distribution refers to a time interval, such as a service time, it is sometimes referred to as a random distribution. This is because, for a time interval that has already begun, each time at which the interval may finish is equally likely (we can often assume that the service time of a server is exponential). It is difficult to give a sound theoretical reason why service times should be exponential, but the fact is that in most cases they are very nearly exponential.

Poisson Distribution

Another important distribution is the Poisson distribution with parameter λ ($\lambda > 0$), which takes on values at the points 0, 1, ...

$$\Pr[X = k] = \frac{\lambda^{k}}{k!} e^{-\lambda} \qquad k = 0,1,2,......$$

$$E[X] = Var[X] = \lambda$$

If $\lambda < 1$, then $\Pr[X=k]$ is maximum for $k = 0$. If $\lambda > 1$ but not an integer, then $\Pr[X=k]$ is maximum for the largest integer smaller than λ; if λ is a positive integer, then there are two maxima at $k = \lambda$ and $k = \lambda-1$.

The way in which the Poisson distribution can be applied to arrival rate is as follows:
If items arrive at a link (connection) according to a Poisson process, this may be expressed as

$$\Pr[k \text{ items arrive in time interval } T] = \frac{(\lambda K)^{k}}{k!} e^{-\lambda T}$$

$$\text{Expected number of items to arrive in time interval } T = \lambda T$$

$$\text{Mean arrival rate, in terms per second} = \lambda$$

Probability of arrival of an item in a small interval is proportional to the length of the interval and is independent of the amount of elapsed time since the arrival of the last item. That is, when items are arriving according to a Poisson process, an item is as likely to arrive at one instant as any other, regardless of the instants at which the other customers arrive.

3.2.1 Generating Traffic

To generate traffic that is proposed in [53][67][80-82], the following steps have been carried out:

1. Generate a series of random number using exponential distribution with parameter λ.

Table 3.1 Examples of Random Number Generated

1.	1	11.	20	21.	118	31.	8	41.	5
2.	4	12.	4	22.	2	32.	10	42.	17
3.	15	13.	9	23.	40	33.	39	43.	29
4.	26	14.	25	24.	43	34.	33	44.	14
5.	35	15.	42	25.	11	35.	17	45.	13
6.	2	16.	5	26.	15	36.	13	46.	41
7.	6	17.	40	27.	41	37.	4	47.	2
8.	1	18.	6	28.	78	38.	2	48.	11
9.	2	19.	5	29.	33	39.	23	49.	14
10.	19	20.	10	30.	12	40.	7	50.	4

Table 3.1 shows some examples of a sequence of random number generated using random exponential distribution with λ = 15. Each random number generated is in terms of 0.1 seconds. To simulate the real environment whereby the connection may become active/inactive (or on/off), the first random number represents an active duration of 0.1 seconds; the second random number represents an inactive duration (0.4 seconds); the third random number represents an active duration of 1.5 seconds and so on. As shown in Table 3.1, the rows with clear background are the active durations while the shaded background are the inactive durations of a connection (e.g. random number 24 is an inactive duration with 4.3 seconds and random number 21 is an active duration with 11.8 seconds).

2. Generate the random arriving packets in all the active durations by using Poisson distribution. Firstly, generate the arriving packets for each active duration (rows with clear background in Table 3.1). The active durations' random number in Table 3.1 will be used as the parameter λ in the Poisson distribution packets arrival generation (e.g. λ = 1 for random number 1; λ = 15 for random number 3; λ = 35 for random number 5).
Table 3.2 shows that 2 packets arrived during the first active duration that lasted 0.1 seconds (the active duration which is generated in step 1). Next, the connection (traffic) will become inactive (with no packets arrived) for 0.4 seconds (0.2 to 0.5 seconds). On 0.6 seconds the connection become active again with an arrival of 15 packets and an arrival of 19 packets on 0.7 seconds and so on.
For clarity, the cells, which are shaded, are the durations where the connection is inactive (0 packets arrival).

Table 3.2 Examples of Random Arrival During Active/Inactive Period

1.	2	13.	11	25.	0	37.	0	49.	36	61.	38
2.	0	14.	12	26.	0	38.	0	50.	31	62.	31
3.	0	15.	21	27.	0	39.	0	51.	46	63.	33
4.	0	16.	18	28.	0	40.	0	52.	30	64.	32
5.	0	17.	17	29.	0	41.	0	53.	35	65.	40
6.	15	18.	12	30.	0	42.	0	54.	40	66.	35
7.	19	19.	14	31.	0	43.	0	55.	38	67.	38
8.	9	20.	13	32.	0	44.	0	56.	21	68.	41
9.	16	21.	0	33.	0	45.	0	57.	40	69.	29
10.	9	22.	0	34.	0	46.	0	58.	35	70.	30
11.	30	23.	0	35.	0	47.	37	59.	32	71.	39
12.	19	24.	0	36.	0	48.	41	60.	37	72.	40

3. These series of traffic will be transformed into binary format for the Neural Network training.

3.3 Constructing a Neural Network for Traffic Predicting

There are many ways in constructing a Neural Network for traffic prediction. The requirement to predict the present value $x(n)$ of a process, given past values of the process that are uniformly spaced in time as shown by $x(n - T)$, $x(n - 2T)$, ..., $x(n - mT)$, where T is the sampling period and m is the prediction order.

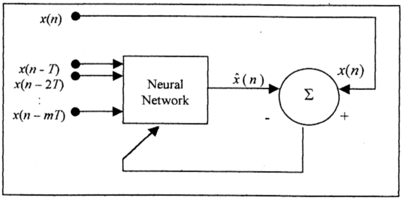

Figure 3.1 Block diagram of non-linear prediction [33].

Using error-correction learning in an unsupervised manner could solve the prediction problem. Since the training examples are drawn directly from the process itself, as illustrated in Figure 3.1, where $x(n)$ serves the purpose of desired response. Let $\hat{x}(n)$ represents the one-step prediction produced by the Neural Network at time n, the error signal $e(n)$ is defined as the difference between $x(n)$ and $\hat{x}(n)$, which is used to adjust the parameters of the Neural Network.

A Neural Network called Adaptive Filtering using Tapped Delay Line (TDL) has been introduced in [43]. This Neural Network is later modified to do prediction. Such TDL is shown in Figure 3.2. The input signal enters from left, and passes through N-1 delays. The output of the TDL is an N-dimensional vector, made up of the input signal at the current time and the previous input signals.

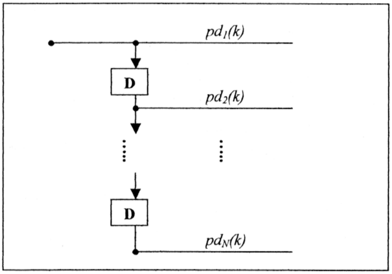

Figure 3.2 Tapped Delay Line (TDL).

This TDL will be combined with conventional Single Layer Feed-Forward Network with single neuron (as shown in Figure 3.3). This network will predict the next value $p(t)$ based on the previous five data. Other notations are defined as follows:

- t – time unit ($t = 1, 2, \ldots.$)
- b – bias
- $p(t\text{-}k)$ – previous k time input ($k = 1, 2, \ldots., 5$)
- i – number of input ($i = 1, 2, \ldots., 6$)
- w_i – connection weight between p_i input and neuron($i = 2, 3, \ldots., 6$)
- $n(t) = \left(\sum_{i=1}^{5} p_i w_i\right) + b$, ($t = 1, 2, \ldots.$)
- $a(t)$ = pure linear transfer function of $n(t)$
- $e(t) = p(t) - a(t)$
- $b_{(t+1)} = b_t + e(t)$, updating bias

- $w_i^{(t+1)} = w_i^t + e(t)p_i$, updating connection weight

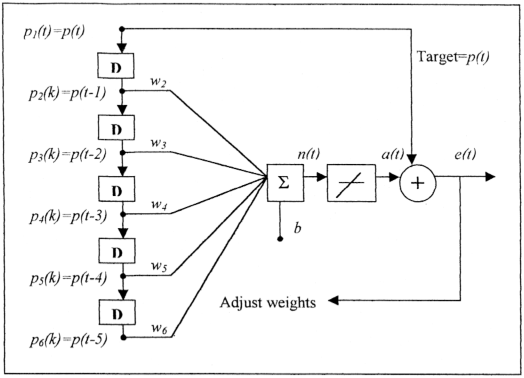

Figure 3.3 Adaptive Filter with TDL for prediction.

The signal to be predicted, *p(t)*, enters from the left into a tapped delay line. The previous five values of *p(t)* are available as output from the TDL. The network is trained to change their weights and bias on each time step to minimize the error *e(t)* on the far right. If *e(t)* is equal to zero, then the network output *a(t)* is exactly equal to *p(t)*. This means the network has done its prediction properly.

3.4 Fuzzy System for Bandwidth Re-allocation

In [47], principal of the proposed model takes three inputs, which are Bandwidth Predicted, Bandwidth Available and Buffer Available. Bandwidth is taken from the under-loading connections and distributed to the over-loading connections. The inputs of the fuzzy inference system are metrics stated in the chapter before. Every metric is fuzzified into linguistic variable for easy reading and understanding. Each linguistic variable is associated with a membership function. In this model, trapezium membership function is chosen for all linguistic variables. Trapezium membership function contains four parameter points (as shown in Figure 3.4, Figure 3.5, Figure 3.6 and Figure 3.7). The parameter points setting are shown in the Table 3.3.

Table 3.3 Inputs, Output and Membership Function

	Membership functions		
Input	Small	Medium	Large
Buffer Available	-37, -0.5, 9.5, 36	9, 45, 60, 91	52, 83, 100, 150
Bandwidth Predicted	-45, 0, 18, 43	9, 45, 60, 91	52, 83, 101, 150
Bandwidth Available	3, 15, 34, 48	33, 48, 70, 80	59, 80, 101, 135
Output	Small	Medium	Large
Bandwidth Allocated	-45, -5, 5, 12	5, 18, 30, 48	38, 70, 105, 145

The three inputs (Buffer Available, Bandwidth Predicted, and Bandwidth Available) and the output (Bandwidth Allocated) have three linguistic variables, which are:

- Small
- Medium
- Large

Figure 3.4, Figure 3.5, Figure 3.6 and Figure 3.7 show the membership functions in diagram presentation.

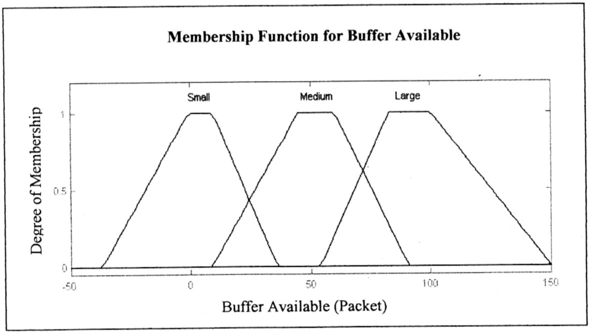

Figure 3.4 Membership function for input – Buffer Available.

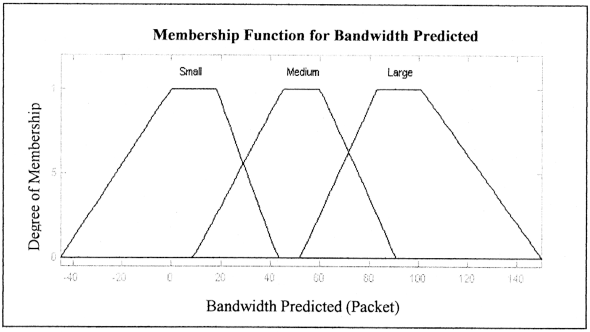

Figure 3.5 Membership function for input – Bandwidth Predicted.

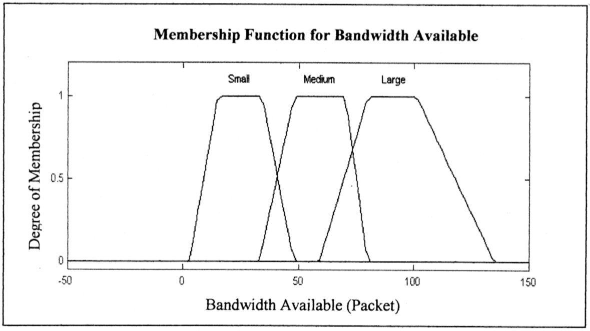

Figure 3.6 Membership function for input – Bandwidth Available.

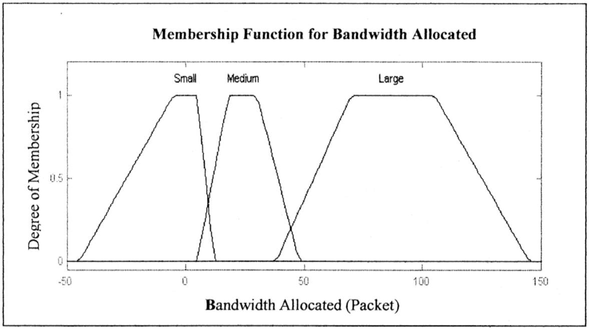

Figure 3.7 Membership function for output – Bandwidth Allocated.

Table 3.4 contains the rule base used in fuzzy inference system. In general, a fuzzy conditional rule contains antecedents and consequents. The antecedents compose by one or more fuzzy predicates. The fuzzy predicates are combined by logical operator AND.

Table 3.4 Rule Base For Fuzzy Inference System

Rule No.	Bandwidth Available	Bandwidth Predicted	Buffer Available	Bandwidth Allocated
1	Small	Small	Small	Medium
2	Small	Small	Medium	Medium
3	Small	Small	Large	Small
4	Small	Medium	Small	Large
5	Small	Medium	Medium	Medium
6	Small	Medium	Large	Small
7	Small	Large	Small	Large
8	Small	Large	Medium	Large
9	Small	Large	Large	Large
10	Medium	Small	Small	Medium
11	Medium	Small	Medium	Medium
12	Medium	Small	Large	Small
13	Medium	Medium	Small	Large
14	Medium	Medium	Medium	Large
15	Medium	Medium	Large	Medium
16	Medium	Large	Small	Large
17	Medium	Large	Medium	Large
18	Medium	Large	Large	Medium
19	Large	Small	Small	Medium
20	Large	Small	Medium	Small

Table 3.4 Rule Base For Fuzzy Inference System (continued)

21	Large	Small	Large	Small
22	Large	Medium	Small	Large
23	Large	Medium	Medium	Large
24	Large	Medium	Large	Medium
25	Large	Large	Small	Large
26	Large	Large	Medium	Large
27	Large	Large	Large	Medium

3.5 Chapter Summary

A method has been proposed and documented in this chapter to solve the dynamic bandwidth allocation problem. Traffic generation method has been previously proposed in [53][67][80-82]. The Neural Network for traffic prediction is taken from [43]. The Fuzzy Logic is a modification from [47] that is originally used for the load balancing application. In the next chapter, the results gathered from the simulation will be presented and discussed.