Load Balancing and Congestion Control in ATM using Fuzzy Logic and Spiking Neuron Network

TAN JIANN SHIN

(WGA99013)

Faculty of Computer Science and Information Technology

University of Malaya

Kuala Lumpur

2005

Load Balancing and Congestion Control in ATM using Fuzzy Logic and Spiking Neuron Network

The thesis is submitted to the

Faculty of Computer Science and Information Technology, University of Malaya,

in partial fulfilment of the requirements for the Degree of Master of Computer Science

by

TAN JIANN SHIN

(WGA99013)

Date: 25 Feb 2005

Supervisor: Professor Dr. Ir. N.Selvanathan

Abstract

The objective of this project is to propose an intelligent routing solution that is capable of increasing network performance in Asynchronous Transfer Mode (ATM) network for Available Bit Rate data traffic. The proposed intelligent routing algorithm, integrates spiking neuron network with fuzzy link cost for the dynamic routing computation.

Although ATM network is a high speed network, it will encounter performance degradation due to heavy traffic or network congestion. Effective intelligent dynamic routing is able to minimize the impact of performance degradation. Among all the introduced concepts on dynamic routing, load balancing is the most popular. Load balancing is proven to be able to increase network performance in various studies [39]. However, distributing network load evenly to load balance network traffic is highly desirable only for single rate traffic. It may not be the best strategy for multi rate environment. Load balancing concept has the tendency to cause bandwidth fragmentation that leads to the result of rejecting connection earlier than it should be. Bandwidth packing is able to minimize the problem of bandwidth fragmentation [34]. Bandwidth packing is a concept of grouping similar traffic which requires similar bandwidth utilization on high utilization links to save bandwidth on other low utilization links for high requirement connections. Both load balancing and bandwidth packing concept may contradict each other, but it is possible to incorporate both concepts into a single routing solution.

Fuzzy logic is introduced to incorporate both concepts (load balancing and bandwidth packing) using the network state metrics (current buffer size, buffer increase rate, bandwidth consumption and requested bandwidth) feedback from each network node. The

capability to simultaneously maximize multiple objectives and easy maintenance makes fuzzy logic the perfect candidate to calculate link cost [5]. Spiking neuron network will use the computed fuzzy link cost to find the minimum cost route. The spiking neuron network is an efficient tool to solve the shortest path routing problem from source node to destination node. It has the flexibility to maximize solution space by recording details of all candidate routes for optimal route selection [37].

An experiment is conducted to measure the performance of the proposed model. A control model is created to work as a benchmark for comparison. According to the experiment result, the proposed model outperforms the control model by accepting more connections while maintaining lower overall cell loss ratio. The proposed model proves that artificial intelligence implementation (using fuzzy link cost and spiking neuron network) realize concept integration easily. The integrated concepts could cover each other's weaknesses and strengthen each other's strengths.

Table of Contents

List of Figures

List of Tables

Acknowledgement

I would like to express my gratitude to the people who made this thesis possible. Special thanks to my supervisor Prof. Dr. Ir. N. Selvanathan for his immense help in guidance and encouragement. With profound knowledge and timely wit from Prof. Dr. Ir. N. Selvanathan in guiding my work, requires no elaboration. His great mentoring is greatly appreciated. My sincere thanks are due to Mr. Ling Teck Chaw and Mr. Phang Keat Keong. Mr. Ling Teck Chaw's confidence and dynamism came as a boon which has greatly given me the boost in my work. Mr. Phang Keat Keong insightful suggestions, company and assurance during the course of work were very valuable and greatly acknowledged. What I know today about the process of research, results from the guidance of Mr. Ling Teck Chaw and Mr. Phang Keat Keong.

My sincere thanks are to my colleagues and friends who have given me much support, interest and valuable hints. Their stimulating suggestions, encouragement and most of all their belief in me have immensely helped me in my research and writing of this thesis.

This page was intentionally left blank

Chapter 1

Introduction

The main objective of this project is to propose an intelligent routing algorithm that increases the network performance in ATM for ABR traffic. The proposed intelligent routing algorithm integrates neural network (spiking neuron network) with fuzzy logic (fuzzy link cost) to perform dynamic routing computation. The intelligent routing algorithm is responsible for planning network resources consumption, avoids using congested resources and avoids the occurrence of congestion if possible. Every network node will contribute their state information such as bandwidth and buffer utilization for route computation. The cell loss ratio and connection acceptance rate are the selected measurement of performance improvement.

1.1 Objective and Scope

The objective of this project study is the search of dynamic intelligent routing algorithm for ATM network. The primary target network traffic is the available bit rate in ATM network. ATM network is having such a need to have a better and effective dynamic routing algorithm for the daily increasing demand of connection from user application. This project study will seek for the possibility to embed artificial intelligent technologies, particularly on neural networks and fuzzy logic, in dynamic routing algorithm for the intelligent routing path selection.

The scope of this thesis is:

- providing details understanding of ATM network and its provided network services
- identifying network state information to be used as parameter for dynamic routing in ATM
- covering on fundamental of neural networks and fuzzy logic, followed by the application and advantages of neural networks and fuzzy logic in dynamic routing
- proposes the feasible implementation of neural networks and fuzzy logic for dynamic routing in ATM network
- conducting a simulation model on the proposed dynamic intelligent routing to verify its practicality and validity
- concludes this thesis with the experiment results as supporting evidences on the proposed model.

1.2 ATM Network

Asynchronous Transfer Mode (ATM) network is a broadband high-speed network for supporting varies integrated services which delivering its services with a guaranteed Quality of Service (QoS). In ATM network, data transmit through network in fixed packet size called ATM cell. Each ATM cell has 5 bytes header and 48 bytes payload. ATM cells are transmitted through virtual path connection (VPC) and virtual circuit connection (VCC). A VPC may consist of a group of VCCs, while a VCC is a concatenation of several VPCs [40]. Note that a physical link may have a few VPCs.

ATM consists of 5 categories of services that have different requirements and characteristics. The services are constant bit rate (CBR), real-time variable bit rate (rt-VBR), non-real-time variable bit rate (nrt-VBR), available bit rate (ABR) and unspecified bit rate (UBR). CBR and rt-VBR are real-time services and the other three are non real-time services. For real-time traffic, the cell arrival time and cell inter-arrival time must fall within a specific tolerable delay time. For non real-time traffic, delay is not a constraint but data loss is not tolerable.

Each request of new connection will invoke call admission control (CAC) to determine its acceptance. A connection is accepted, if ATM network is able to satisfy the requested QoS. Meanwhile, usage parameter control (UPC) will enforce the agreement between user and ATM network to guarantee the agreed QoS [40].

Although ATM network is a high-speed network, network congestion that would degrade its performance is still possible. The occurrence of network congestion is primarily due to unlimited request on limited network resources especially on network's "bottleneck" or "hot spot" area. Hence, effective network resources planning to minimize the effect of network congestion, is the center of focus in this project.

1.3 Network Congestion & Dynamic Routing

Network congestion is a state in which network performance degrades dramatically due to saturation of network resources such as connection bandwidth, processors time and memory buffers. The effect of network congestion includes delay in data transmission and waste of resources usage. If appropriate action is not taken quickly, the congestion

phenomenon will spread quickly to other adjacent nodes and eventually bring down the entire network. Therefore, congestion control is introduced to solve this problem.

Congestion control is becoming one of the crucial issues in the design and research area of computer network. This is due to the continuous increasing demand of intensive applications that requires more bandwidth and higher quality. The existing approaches for network congestion control can basically be divided into two categories: congestion avoidance and congestion recovery. The congestion avoidance approach is to maintain the workload of network near the knee point without exceeding it, so that congestion will not occur. Knee point is where the increase of throughput becomes much slower than the increase of the load. When the network load exceeds the knee point, the network congestion phenomena will begin. The congestion recovery approach is to recover the network operation whenever the network detects network congestion. For ATM network, the existing congestion controls include credit-based control scheme [18], rate-based control scheme [47] and balanced control scheme [53]. Figure 1 shows the throughput-load relationship in computer network without congestion control [30]. Area A is the area that has no congestion, area B is moderate congestion and area C is severe congestion.

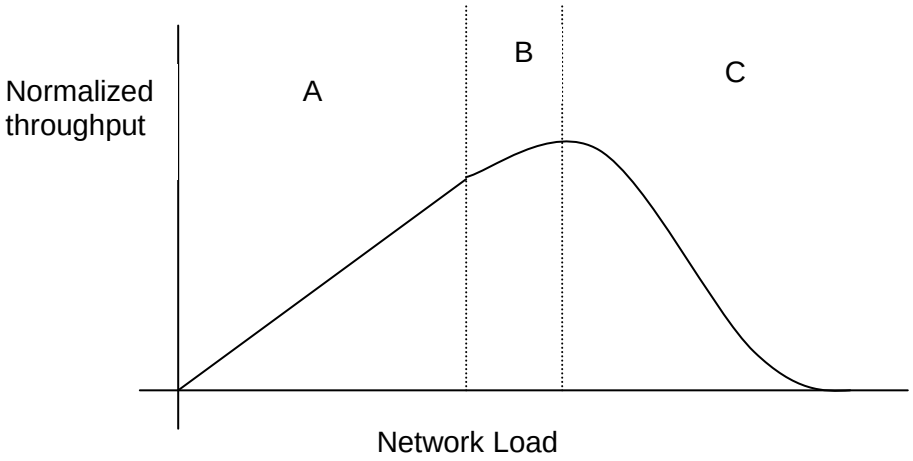

Figure 1: Throughput-Load

source: [30]

Most of the existing congestion control scheme is based on reducing the transmission rate from source. However, this technique is not always applicable especially in ATM network traffic. ATM network traffic except UBR traffic has its guaranteed minimum transmission rate where transmission rate control is not suitable. Dynamic routing opens a new solution space for congestion control by adopting the concept of load balancing or bandwidth packing. Load balancing is a concept that distributes network load to other nodes that has the same capability in the network. This could help to relief the network load in heavy loaded network nodes to achieve higher cost effectiveness [39]. Bandwidth packing is a concept that tries to utilize 100% of used connection links before considering the unused connection links. Employing bandwidth packing concept helps to avoid bandwidth fragmentation problem [34].

Network congestion always occurs at over loaded network nodes and links. While some network nodes and links are suffering from the network congestion, some of the network nodes and links are actually under-loaded. The situation of the network begins rejecting

connection request while having available network resources is something network operators do not wish to happen. This is highly undesirable in ATM network because the network facilities and equipments are very expensive.

Load balancing is proven able to maximize the overall utilization of computer network by moving some of the workload from over-loaded network nodes to under-loaded network nodes. This mechanism delays the occurring of congestion and reduces delay time. Indirectly, adopting load balancing in ATM network is able to increase connection acceptance rate [39].

Bandwidth packing is another concept that is proven to be able to maximize computer network overall utilization. Bandwidth packing achieves the result by first considering used links to accept new request until it reaches 100% utilization before using the unused links. The bandwidth packing policy on saving unused links for higher requirement connection has successfully increased the connection acceptance rate [34].

All routing schemes have the following underlying principles [42]:

- Assembling and distributing state information for network traffic

  Network state information may include provided services, existing load, available resources, and abnormal conditions in the network. Network state information are measured at selected nodes or each node.

- Selecting feasible or optimal routes based on network state information

  Relying on the gathered network information, a network element shall determine the feasible or optimal routing path for a connection. Feasible routes are those that satisfy the expected service requirements. Optimal routes are subset of feasible routes that are the best route compliance with the network routing policy.

Computing an optimal route for a connection often requires intensive resources and is difficult to achieve when there are too many factors influencing the selection.

- Forwarding traffic along selected routes

  Once a route has been selected, the network traffic of the connection can be forwarded according to one of these two forwarding paradigms: connection-oriented or connection-less forwarding. Connection-oriented forwarding requires connections to be set up prior to use the route to transmit data. Connection-less forwarding requires that the transmitted data carry enough information to reach the destination.

As stated above, dynamic routing system will refer to the gathered network state information such as buffer and link utilization in route determination process. The network state information are gathered through prediction using history records or real time measurement. Selecting the proper network state information is crucial to enable computer network to response accurately to the changes within the network especially network congestion. A number of dynamic routing schemes are proposed for ATM networks. Details of the routing schemes and their selected network state parameters are discussed in Chapter 2.

ATM routing is considered as an instance of Non-deterministic Polynomial complete problem (NP-complete problem) as any additional factor could cause its routing computation time to increase exponentially [29]. A NP-comlete problem is a problem that at least one instace of the problem cannot be solved in polynomial time. In ATM routing many factors are taken into consideration and requires some trade off to find an optimal route within a reasonable time. Implementing artificial intelligence into dynamic routing algorithm allows the routing decision to mimic the network operator’s decision and

avoiding ATM routing becomes NP-complete problem. With artificial intelligence within the routing algorithm, such dynamic routing could refer as the intelligent routing algorithm.

1.4 Fuzzy Logic

Fuzzy logic is a concept introduced by Prof. Lofti Zadeh in University of California, Berkeley. Fuzzy logic provides a simple way to arrive at a definite conclusion using some vague, ambiguous, imprecise, noisy or missing input information. It allows partial set membership rather than crisp set membership or non-membership. Membership is the belonging measurement of a classification. A crisp set membership is a definite classification set theory. Fuzzy logic is a simple rule-based solution capable to solve control and interpretation problem. It requires some numerical parameters to operate but the exact values of these parameters are usually not critical. In the proposed solution, fuzzy logic is calculating the cost on each link. The calculated cost, hereafter refers as fuzzy link cost is then used in route determination process. Chapter 2 will discuss the study and application on fuzzy logic. The proposed fuzzy link cost implementation is in Chapter 3.

1.5 Neural Network

Neural network is an artificial intelligence technique imitating the human nervous system that contains a collection of neuron units communicating with each other via axon connections. Neurons and the interconnection synapses constitute the key elements for neural information processing. Neuron is the information processing unit in neural network. A set of synapses is connected to neurons with each synapse characterized by a weight or strength. Artificial neural network normally go through a learning process before it become

stable. During the learning process, there are no obvious changes in its parameter value or result matching the target data, except in supervised neural network. In each cycle of learning, each synapse (connected link) adjusts its weight through a corrective adjustment function using feedback as guideline. Spiking neural network is a new type of neural network model that employs spiking neurons as computational units. The spiking neuron (also known as “integrate and fire neuron”) is using time as resources for computation and communication. Chapter 2 will discuss neural network studies and applications in ATM. The implementation of the proposed spiking neuron network is in Chapter 3.

1.6 Motivation and Document Organization

The motivation for study and solving network congestion phenomena through intelligent routing is to efficiently increase the overall network resources utilization while maintaining the guaranteed quality of service. This is important for application that requires relatively high quality of data transmission. Tele-medicine application is one of the applications that may need high quality of service with low delay time. The application may be real time videoconferencing application, interaction audio application or large file transfer. Besides providing better services, from the economical point of view, the overall network resources should achieve maximum utilization to increase cost-effectiveness.

The scope of this thesis studies the problem domain of dynamic routing and simulates the proposed intelligent routing solution. ATM network, dynamic routing, neural network and fuzzy logic are the major research areas. For ATM network switching mechanism, Connection Admission Control (CAC) and Usage Parameter Control (UPC) are not within the scope. The ATM information gathering mechanism and existing ATM network dynamic routing schemes are included in the study. Analysis on neural network and fuzzy logic

applications helps to understand the integration of both artificial intelligence techniques. The proposed algorithm with artificial intelligence implementation, spiking neuron network and fuzzy link cost, will test run on ATM simulator to examine its performance.

The next chapter will go through related researches within this project's scope. The study areas include the ATM network, dynamic routing, neural network and fuzzy logic. The detail design for the proposed solution will be presented in Chapter 3. Chapter 4 discuss on simulation results of the proposed solution. Chapter 5 concludes the thesis and possible future improvement.

Chapter 2

Literature Review

The previous chapter introduces the objective and focus of the study area. This chapter will discuss on papers related to the study area such as dynamic routing, fuzzy logic and neural network. The integration of fuzzy logic and neural network opens a better solution for solving dynamic routing as intelligent routing in ATM network.

2.1 ATM Network

ATM (Asynchronous Transfer Mode) also known as cell relay network is a broadband high-speed network supporting varies integrated services. ATM allows multiple logical connections to be multiplexed over a single physical interface. In ATM networks, data is transmitted through the network, in fixed packet size named ATM cell where each cell has 5 bytes header and 48 bytes payload. The use of small fixed-size cell provides the following advantages [40]:

1. It simplifies the required processing on each ATM cell that helps to support the use of ATM at high data rate.
2. It is an easy implementation of switching mechanism in hardware.
3. It reduces queuing delay for a high priority cell because it waits less if the high priority cell arrives slightly behind a lower priority cell that has gained access to a resource.

2.1.1 ATM Header

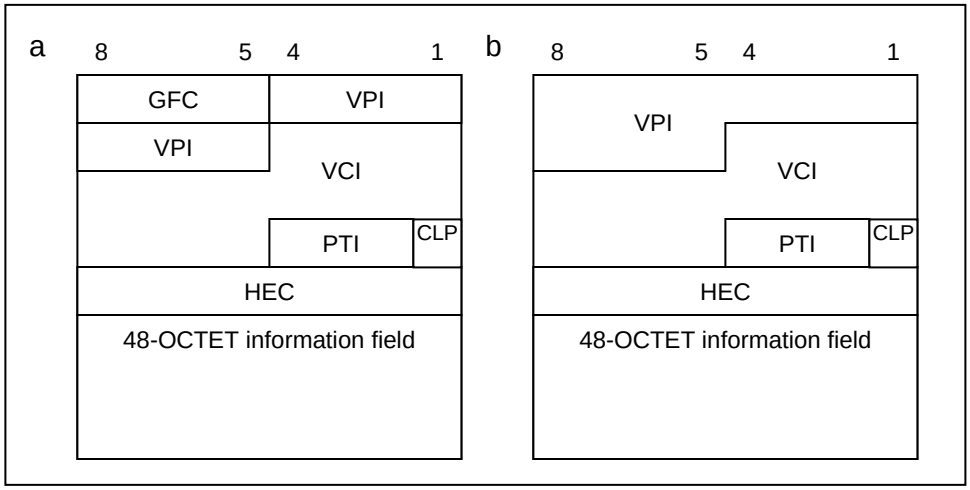

Figure 2: ATM cell format a) user-network interface, b) network-network interface

Each header field of an ATM cell has its own characteristics and function. It is versatile in the sense that the ATM switch will alter header field when the cell is transferred in the network. The header fields of ATM cell are [40]:

- Generic flow control (GFC) – Generic flow control is a 4 bits field. This field is introduced to alleviate short-term overload condition in the network by controlling the entry of cells into the network. Generic flow control will exist in the header only when the cells are transferred over user-network interface. This field becomes part of VPI field when the cells are transferred over network-to-network interface.
- Virtual path identifier (VPI) – Virtual path identifier is an 8 bits field when the cell is used in the user-network interface but expands to 12 bits by utilizing the GFC field in network-to-network interface. This field is used for identification and routing within the network.

- Virtual circuit identifier (VCI) – Virtual circuit identifier is a 16 bits field. This field is used for identification and routing within the network. Combination of VPI and VCI gives the results of a unique identification.
- Payload type identifier (PTI) – Payload type identifier is a 3 bits field. This field is an information type indicator in the information field. The following is the description for each bit:

    i. The first bit is used to identify the cell type. Value 0 in the first bit indicates user information while value 1 indicates network management information.
    ii. The second bit indicates the congestion state of the network. Value 0 indicates no congestion and value 1 for congested network.
    iii. The third bit is used to indicate the beginning, continuation or the end of data. Value 0 is for beginning or continuation of data transfer while value 1 is for the end of data.

- Cell Loss Priority (CLP) – Cell loss priority is a 1 bit field. This field provides guidance to the network in the event of congestion. A value of 0 indicates a relatively higher priority cell, which the cell should not be discarded unless no other alternative is available. A value of 1 indicates that this cell is subject to be discarded within the network. The user might employ this field so that extra cells (beyond the negotiated rate) may be inserted into the network with a CLP of 1, and deliver to the destination if the network is not congested. The network may also set this field to 1 for any data cells that are in violation of the agreed traffic.
- Header error control (HEC) – Header error control is an 8 bits field. It is an 8 bits CRC on the first 4 octets of the header.

ATM network is capable of transmitting data at the speed of 155.52Mbps and 622Mbps that classifies itself as high-speed network. In the architecture of ATM network, it consists of 3 layers: physical, ATM and ATM adaptation.

- Physical layer: The physical layer is the physical connection between ATM switches that perform physical data transmission.
- ATM layer: ATM layer is a common layer to all ATM supported services by providing general cells transfer capabilities. ATM data transmission is based on the concept of virtual path connection and virtual circuit connection.
- ATM adaptation layer: ATM is designed to support multiple types of services (constant bit rate service, real-time variable service, non real-time variable service, available bit rate service and unspecified bit rate) that have different characteristics and requirements. Almost every supported service has its own adaptation layer to provide specific handling.

2.1.2 ATM Layer

ATM layer is providing a common cell transfer mechanism to all services by having the ATM adaptation layer bridging the gaps. ATM layer handles the logical connections using the concept of virtual path connection and virtual circuit connection. The following figure shows the implementation of virtual path connection and virtual circuit connection.

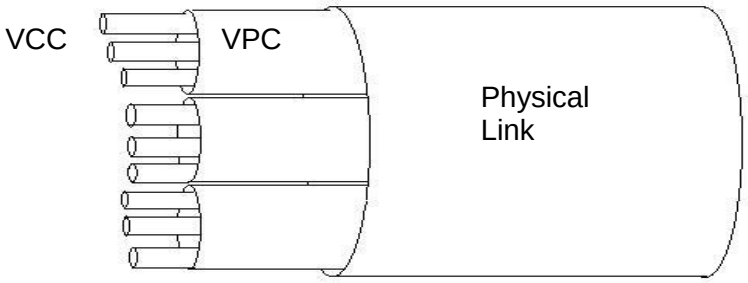

Figure 3: Virtual Path and Virtual Circuit Implementation

Virtual Circuit Connection, VCC is the basic unit of logical unidirectional connection in ATM network. All data transmissions are carried out on the VCC. The use of VCC could be as follows:

1. Connection between 2 end users for data exchange,
2. User-to-network connection for control signaling and
3. Network-to-network connection for network management and routing.

Virtual Path Connection, VPC is a bundle of VCC that have the same exit point. Thus, all cells flowing over VCCs in a single VPC are switched along the same path. Every VCC must reside within at least 1 VPC. In certain case, a VCC may go through 2 VPCs by concatenating the VPCs to reach the destination node. The concept of using VPC is in the response to a trend in high-speed network, where the network control costs are increasingly higher in proportion to the overall network cost. The employment of VPC helps to contain the control cost by grouping connections sharing a common path into a single unit. The employment of VPC brings the following advantages to the network operators:

1. Increase network management efficiency by controlling a small group of connections only instead of a large number of individual connections.
2. Increase network performance by reducing processing time and shorten connection setup time. The underlying network path is done when a virtual path is setup. A new VCC can be established easily with no additional processing at intermediate nodes by executing simple control functions at the end point of the VPC.
3. Enhanced network services such as defining closed user groups or closed network connections.

The process of setting up a VPC is decoupled from the process of setting up a VCC. A VPC to the destination node with sufficient available bandwidth must first be ready before setting up a VCC. However, a VPC does not require having any VCC running through it all the time. The occurrence of such situation is usually due to the VCC has completed its data transfer. The VPC setup mechanism includes calculating routes, allocating bandwidth and storing connection state information. In all cases, cell sequence integrity is preserved within a VCC as cells are delivered in the same order as they are sent. Every virtual path and virtual circuit has a unique numerical tag namely virtual path identifier (VPI) and virtual circuit identifier (VCI) that help to identify the virtual path and virtual circuit. Since the virtual circuit is contained within a virtual path, the virtual circuit identifier is unique in a virtual path only. In order to uniquely identify a VCC connection in the ATM network, a pair of VPI/VCI is used as reference.

2.1.3 ATM Services

ATM is designed to be able to transfer many different types of traffic simultaneously including real-time flows and bursty TCP flows. Due to the diversify characteristics and requirements of the traffic flow, each service is handled by specific adaptation layer. There are two main categories of service, real-time services and non real-time services. Real time services consist of constant bit rate (CBR) and real time variable bit rate (rt-VBR). For non real-time services, it consists of non real-time variable bit rate (nrt-VBR), available bit rate (ABR) and unspecified bit rate (UBR) [40].

2.1.3.1 Real-Time Services

The most important measurement of real-time services quality is the delay and the variation of delay. Data of a real-time service application must be delivered within the application tolerable time to maintain its quality. Lacks of continuity or excessive data loss will result in significant low quality for real-time services. Applications that involve human interaction are having tight constraint on the delay, as the delay will negate the functionality of the application. In such high requirement applications, any delay above a few hundred milliseconds becomes noticeable and annoying to the users. The following are the available real-time services supported by ATM network:

- Constant Bit Rate (CBR)

  Constant bit rate is used by application that requires a fixed data rate to continuously stream data within a short transfer time for the entire connection

lifetime. CBR is commonly used for uncompressed audio and video information. Its application includes: videoconferencing and interactive audio.

- Real time Variable Bit Rate (rt-VBR)

  Real time variable bit rate is introduced for time-sensitive application with high requirement on delay and delay variation. The principal difference is to choose the appropriate real-time service for an application, is the requirement on data rate. For example, a real-time video requires a uniform frame rate to avoid jitter, but its actual data rates are variable. rt-VBR is not constant using a fix amount of bandwidth which allows the network to multiplex a number of connections statistically over the same dedicated capacity while maintaining guaranteed quality of service to each connection.

2.1.3.2 Non Real-Time Services

Non real-time services are intended for applications that do not have any time constraint. This gives the network greater flexibility in handling traffic flow and can make greater use of multiplexing to increase network efficiency. Some of the non real-time services are having bursty traffic while some of them are data loss sensitive. Therefore, specific adaptation layers are design for these services.

- Non real-time variable bit rate (nrt-VBR)

  Some non real-time applications are possible to characterize their expected traffic flow which can allow the network to utilize these information to improve quality of service in the area of data loss and delay. The traffic characteristics (peak cell rate, average cell rate and how bursty the traffic will be) are provided by the end system application to the ATM network. The ATM network allocates sufficient resources to

the connection based on the provided characteristic. This service is suitable for transferring application that has critical response-time requirements such as airline reservations and banking transaction.

- Available bit rate (ABR)

  Available bit rate is introduced for bursty application that uses a reliable end-to-end protocol for congestion detection. The ABR flow control mechanism uses an explicit feedback to the source node to perform flow control. Applications using ABR specify peak cell rate (PCR) and minimum cell rate (MCR) to the network for the allocation of at least MCR capacity during the connection lifetime. Any unused capacity in the ATM network is shared in a fair and controlled fashion among all ABR connections. Any capacity not used by ABR will then be allocated for unspecified bit rate traffic.

- Unspecified bit rate (UBS)

  Unspecified bit rate is a service that makes use of any remaining available bandwidth in the ATM network. This service is only suitable for application that could tolerate delays and some cell losses. At any given time, certain amount of the ATM network is always carrying CBR and VBR traffic. However, there are available bandwidths for UBR service for one or both reasons below:

  a. Not all of the total resources are committed to CBR and VBR traffic
  b. The bursty nature of VBR traffic means that at some period of time it uses less than the committed capacity

  No commitment is made to UBR connection and no feedback regarding network congestion is provided which leads to the high possibility of variable delays and data losses. Hence, UBR service is also known as best-effort service.

2.1.4 ATM Adaptation Layer (AAL)

ATM adaptation layer is needed to support information transfer protocol not based on ATM network. ITU-T has defined 4 classes of services that cover a broad range of protocols. The classification is based on the application traffic flow characteristics such as data rate, connection oriented or connectionless. In general, an AAL layer is organized in 2 logical sublayers:

1. Convergence sublayer (CS) – convergence sublayer provides the convergence function between the service offered and the underlying ATM layer. Since each AAL is designated for different type of services, different CS protocols are associated with each AAL and its service access points.
2. Segmentation and reassembly sublayer (SAR) – Segmentation and reassembly sublayer is responsible for packaging information received from its upper layer (convergence sublayer) into cells for transmission and unpacking the cells into information data received from its lower layer (ATM layer) for application use. Similarly, there are different types of SAR protocols with their own structure.

The 4 classes of service are [40]:

1. AAL 1 – AAL 1 is used for dealing with constant bit rate source. Each cell of this AAL is embedded with a sequence number to track lost sequence.
2. AAL 2 – AAL 2 is intended for handling analog applications such as video and audio that has timing relationship but does not require constant bit rate. The application timing relationship is on the frame rate instead of data rate, which also means that the data rate is a variable.

3. AAL3/ 4 – AAL 3 and AAL 4 are very similar in terms of their functionality and packet data unit (PDU) format. Therefore, ITU-T decided to combine these two AAL layers together become AAL3/ 4. This AAL could service connection oriented or connectionless protocols where its convergence sublayer handles the adjustment for all supported protocols. The distinct feature that could only be found in AAL 3/ 4 is its capability to multiplex different streams of data on the same virtual ATM connection (VPI/VCI).
4. AAL 5 – AAL 5 is introduced to provide a streamlined transport facility for connection-oriented protocol. This AAL is designed to adapt to existing transport protocol while reduces protocol processing overhead, transmission overhead by having protocol header in only the first transmitted cell. In the subsequent transmitted cell of this AAL contains only ATM header and the data packet.

2.2 Dynamic Routing

Effective dynamic routing is a “wicked” problem that exists in computer network. The effectiveness of the routing algorithm relies on its adaptability on traffic and network state such as congestion and bursty traffic. The information gathered for dynamic routing will easily become “outdated” cause by network delay. Outdated information will render the dynamic routing algorithm ineffective and may even paralyze the entire network [7][12][14][20][40][42][43][53]. The difficulty increases when the routing algorithm plans for multiple categories of service with different requirement. Routing efficiently in ATM while maintaining its Quality of Service is a multiple constrains shortest path problem where some constrains may contradicts each other. The contradiction of constrains will lead to a NP-complete problem [29]. A NP-complete problem is a problem that at least one instance of the problem cannot be solved in polynomial time.

2.2.1 ATM PNNI

In network routing, obtaining topology information and maintaining up-to-date information for each network node is important. The network information gathered by each network node will determine the accuracy and efficiency of implemented routing algorithm. Two main routing approaches exist, distance-vector routing and link-state routing. Distance-vector routing distributes the routing information by sending the entire network node routing table to its neighboring nodes. The receiving nodes then update their own routing table. The process will repeat itself to propagate the routing information. Link-state routing distributes the network information by broadcasting the status of the network node to all network nodes. The receiving nodes then use the information to maintain their network topology. The distance-vector has the disadvantage of taking a long time to update the entire network whenever there is a failure in the network. Although link-state routing is faster in response to network changes, link-state routing suffers from heavy computation and storage problem when the network becomes big.

Private Network Node Interface (PNNI) is a hierarchical, link-state routing protocol proposed by ATM Forum that organizes ATM switches into logical groups called peer groups. Switches identify their peer group by using the Peer Group Identifier (PGID). Switches that belong to the same peer group have a common PGID. Nodes that belong to the same peer group become aware of each other via the exchange of Hello packet. Each node will exchange its database information using PNNI Topology State Elements (PTSE) that contains topology characteristics derived from the link or node state information. PTSEs are grouped to form PNNI Topology State Packets (PTSP). The PTSPs are flooded within each peer group. Every node in the same peer group will have identical

databases. Each peer group will elect a peer group leader, which is responsible for aggregating information and distributing aggregate information to the higher layers. The process of grouping ATM switches and distributing aggregated information is repeated for every higher level. The lowest level switch physically implements the functionality of distributing aggregate information for higher level switch [7].

In PNNI, at the lowest level of the hierarchy, each node represents a physical ATM switch with a unique ATM address and each link represents a physical link or ATM virtual path. In the higher level, each node represents a collection of lower level nodes and each link represents a collection of lower level links [14].

Call establishment in PNNI consists of two operations - path selection and connection setup at each point along that path. PNNI employs source routing for its path selection. The first switch for a given connection is responsible to select an optimal route, based on network topology and network load in its database for the connection. Djikstra's algorithm is mentioned for computing minimum cost route based on link metrics in Appendix H of the PNNI standard. The route is encoded as a stack of Designated Transit Lists (DTLs), which is explicitly included in the setup signaling message [7].

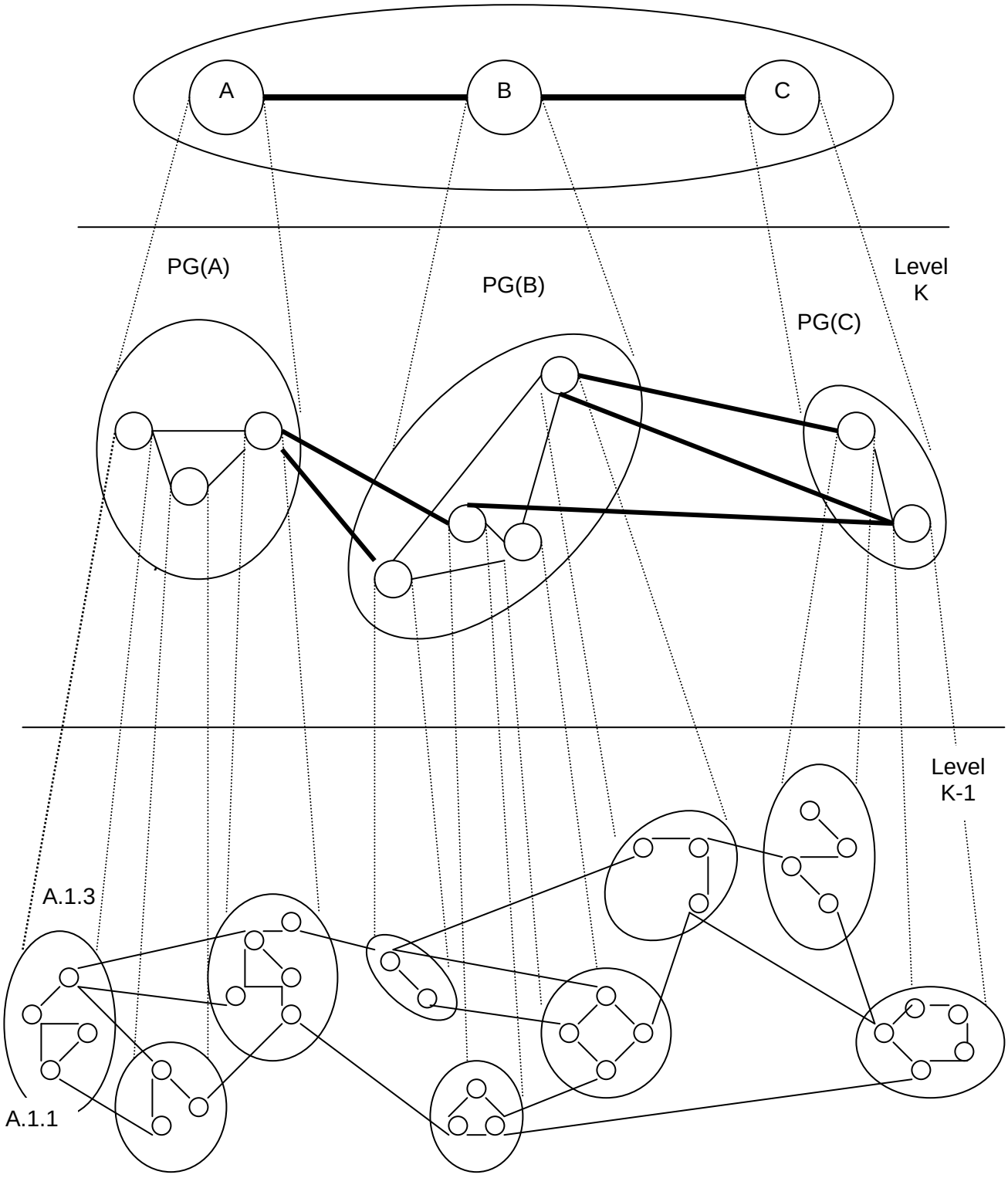

Figure 4 Example of ATM PNNI hierarchy

Many studies on using PNNI in ATM network routing are conducted by the inspiration from the ATM Forum. In [43], this paper discusses hierarchical routing using PNNI in ATM network. This paper argues that any routing problem that has a solution in a single layer network could be applied to the multi layers hierarchical network and the main challenge in hierarchical routing is the construction of hierarchy than the routing algorithms itself. The reason for its finding is due to the information condensation that leads to the inaccuracy during route computation which depends on the methodology used to calculate the hierarchical structure. This paper employs the basic principle of routing hierarchical network which first search for a level k that consists logical representation node of source A and the logical representation node of destination B in one logical subnet. A routing path is then constructed at the level k from logical source A to logical destination B. The routing algorithm will then specify the path in greater detail by descending to a lower level until reaching the lowest level which has the physical network node and links. The result of this routing algorithm is returned as a destination transit list which represents the physical path from source A to destination B. However, in the research of [14], the paper indicates that some of the network nodes – mainly those that function as border node – may be overloaded with route computation processes, while other nodes are rarely involved. The same hierarchical routing algorithm of [43] is applied in the research of [14]. The paper uses the “source routing” scheme in which the routing computation is done at the source node. When a routing path is needed, a less detailed path is calculated at level k and as the routing algorithm descend to a lower level another detailed path is recomputed by each border node for each involved peer group in order to determine the best route to cross a peer group. A border node may then involve a repetitive route computation for a single routing path. Therefore, the paper [14] proposes a scheme to balance the computational burden among the network nodes by transferring some of the computation task to other under-utilize network nodes.

In certain cases, a network node in the selected path may not have sufficient resources to accommodate the connection even though the source switch selects that path. This may occur if the source switch has “old” information or suppressed information due to hierarchical aggregation. In such cases, crank back process will occur where a connection cannot be accommodated at the switch because of CAC failure. The connection will crank back to the source switch for an alternative route determination. This is obviously an undesirable action since it delays the connection setup procedure.

Multipath routing combined with resource reservation proposed by [9] suggests reserving resources along several routes in parallel during connection setup. The experimental results have shown that the suggestion increases network utilization and have faster connection setup time.

This paper [9] suggests that during the connection setup, resources of a link are reserved whenever it is considering the link for the connection. If the route fails to establish at the network node, the resources downstream of the network node are released and crank-back to an immediate node. The advantages of implementing this routing algorithm are:

1. This algorithm is able to choose the best route among all successful established routes for better utilization.
2. Faster connection setup time as several routes are established in parallel that helps to minimize or eliminate the delay cause by crank-back mechanism. However, the penalty is that redundant resources are locked during the connection setup time.

2.2.2 Bandwidth Packing and Load Balancing

Besides PNNI, there exist a few different approaches of routing algorithms for ATM networks. The most popular and widely used for comparison is the least loaded routing [35]. Least loaded routing algorithm always select the least utilized candidate route with the condition it could support requested bandwidth. In *Performance Studies of Dynamic Load Balancing in Distributed Systems* [39], load balancing was found to be able to increase network performance significantly, especially under heavy or unbalanced workloads.

Distributing network load evenly is highly desirable for single rate traffic, but it may not be the best strategy in multi rate environment. Modified least loaded routing proposed in [20] for virtual path based ATM network is a dynamic routing algorithm that implements bandwidth packing concept with load balancing. This routing algorithm supports multiple traffic classes with varying traffic characteristic.

The original least load routing is the most popular dynamic routing scheme to achieve the load balancing objective. Using original least load routing, whenever a connection fails to establish through minimum hop route, alternative route with the largest available bandwidth is selected. For multiple classes network traffic, pure load balancing might cause bandwidth fragmentation that leads to early call blocking. The proposed modified least loaded routing functions are as below:

1. A class k call is given direct route whenever the call is acceptable.
2. If direct route cannot accept this call, alternative routes will then be offered.

3. If there is a set of alternative route for class k, adopt load balancing concept by selecting the route with largest available bandwidth.
4. If alternative routes are on route not for class k, select routes carrying calls with smallest required bandwidth per call. If there are more than one candidate routes, select the route with largest available bandwidth. The rational behind the selection of routes carrying calls with smallest required bandwidth per call is that these routes are potentially able to minimize the effect of bandwidth fragmentation.
5. If no alternative routes then block the call.

In [20], the considered alternative route is 2 links alternative route only. For example, the alternative route for direct link between switch A and switch B, is switch A to switch C to switch B. The reason for considering 2 links alternative route is that it consumes more resources when using more links alternative.

In *Performance Modeling and Management of High-Speed Networks* [34], it has been recognized that in order to maximize the available resources utilization, a routing policy in a heterogeneous multirate environment should implement packing of narrowband VCs (having relatively small bandwidth requirement) on selected paths in order to leave room on other paths for wideband VCs (having relatively large bandwidth requirement). Bandwidth packing strategy achieves two desired results:

1. Helps to minimize the problem of bandwidth fragmentation that leads to higher utilization,
2. Improved fairness between narrowband and wideband connection by increasing the acceptance of wideband connection.

Load profiling routing scheme proposed by [8] is a concept that distributes network load to a set of candidate routes based on the connections' characteristics. This routing scheme is a variation of bandwidth packing concept. It uses the route probabilistic selection to determine the route whereby it calculates the probabilities of available bandwidth distribution on candidate routes to match the multi-class demand traffic distribution. As a result, this routing scheme will try to utilize 100% of used path to save the unused path for connection with larger bandwidth requirement. By saving the unused path helps to increase the likelihood of succeeding in accepting new request. The experimental results demonstrate that the acceptance of connection increases by maintaining bandwidth availability profile especially when a request with high capacity requirement is submitted to the network. This paper argues that load balancing is not the primary goal when performing multiple classes of traffic routing. The more important goal is to increase the accepted chance of future incoming request.

From *Dynamic Routing Algorithms in VP-based ATM Networks* [20] and *Load Profiling for Efficient Route Selection in Multiclass Networks* [8], it is clear that bandwidth packing and load balancing do help increase the overall network utilization. Although both concepts are contradicting, incorporating both concepts in a single routing scheme is possible and may further increase the overall network utilization.

Dynamic routing algorithm proposed by [15] is implemented by manipulating VP connection such as replace obsolete VP with new VP connection. When a requested connection cannot be met at local node, the request is sent to central node where the dynamic VP routing algorithm is employed to attempt to establish a new VPC. The dynamic VP routing algorithm will minimize the variance of bandwidth utilization and the route length.

The author of [16] proposes a complete ATM network management solution including routing management. The routing management employs load balancing with 2 levels of hierarchies:

- The higher level, routing plan component, (operates at epochs where network usage predictions change) produces new sets of routes based on the current set of VPCs.
- The lower level, load balancing component, (operates within time-frame of network usage predictions) based on the actual network usage selects optimal route from a set of routes established in the higher level.

The proposed algorithm is built based on the concept of 'route potentiality' for setting up new connections. The route potentiality is a function to calculate the possibility of setting up the route by referring requested bandwidth, available bandwidth and the alterability of VPC.

Both [15] and [16] are motivated by the ATM network management solution states that the management component or central node could improves routing in ATM network. The management component or central node with more information could have better "planning" to advise the creation of connections. By employing help from management component or central node, this will relief the border node from heavy burden of route computation as stated in [14].

Linear-based trunk reservation routing algorithm proposed in [6] employs the trunk reservation concept. It is done according to a probability value that increases linearly to its maximum value (equal to 1) as the traffic load increases. Referring to *A Linear-based Trunk Reservation Routing Algorithm for ATM Networks* [6], during the light traffic load with plenty of resources available, an algorithm should try to balance the load effectively to utilize the unused resources by using alternative route. However, as the network load

becomes heavy, it is undesirable to use alternative route that consumes more network resources. Reserving minimum hop route in heavy load network for better resources usage results the increase of calls acceptance which leads to higher network throughput. Thus, whenever there is a new call request, the routing algorithm should consider the required resources and the congestion level of each route to make an effective trade-off between minimum hop and load balancing.

The algorithm is implemented using a variable bandwidth utilization threshold, P_{trunk}. When the expected route utilization, P_{route} (which is defined as the maximum expected bandwidth utilization), is bigger than the threshold, only minimum hop route is allowed to use the link. The value of P_{trunk} is suggested to increase smoothly for better performance by adapting to the network load changes.

Its effectiveness is based on the fact that the saving of network resources is very critical at heavy loaded network. The high utilization links are selected only for minimum-hop routes, resulting in better network resources usage. However, it has negative effect on its performance at moderate and light loads. The reason is the small P_{trunk} value blocks the minimum hop route earlier and leads to using up the alternate routes earlier, resulting in increased connection blocking rate at light load (when the minimum-hop routes are blocked, alternate routes cannot be used). When the network load increases, P_{trunk} value increases too. Increased of P_{trunk} value opens reserved network resources on the minimum hop route to accept connections, which leads to increase of call acceptance rate at heavy loaded network.

2.3 Neural Network and Fuzzy Logic in ATM

2.3.1 Fuzzy Logic

Similar to any study of artificial intelligence, fuzzy logic is emulating the expert human operator by going through the fuzzy rule set in making decision or solution finding. For most of the problems, particularly controlling problem, the important information that will influence the solution are from two sources: the sensory measurement capturing the reading of selected parameters in the system in numeric format and human expert knowledge that describe in words and natural language. The reading of the selected parameters reports the current status of the system and act as an indicator for the system to react. Meanwhile, the human expert knowledge is the guideline or reference for the system to react or to calculate a solution in order to change the system into its next desired state. The human expert will record all of their knowledge about the system as actions to be taken associating with conditions. These actions with conditions are also known as the system rules or criteria for solution finding. Both sources of information are critical to the system functionality as the sensory measurement indicating the current state of the system and the human knowledge moves the system into its next desired state. An excellent solution finding methodology should be able to assimilate the information from both sources [26] and [38].

In classical control theory, the human knowledge is transformed into a mathematical model for solution computation and to suite into the sensory information dimension. In the case of linear problem, transforming human knowledge into mathematical model is never an issue. However, in the case of nonlinear problem, the mathematical model would normally be

complex and difficult to formulate. The situation becomes worse when the complexity of the system increases.

Fuzzy logic is contrary with the classical control theory. Instead of transforming the human knowledge from natural language into mathematical model, fuzzy logic fuzzifies the sensory numeric information into natural language. The concept of linguistic variable describes the sensory information in fuzzy sets. Since the human knowledge is preserved in natural language as fuzzy rule set, a complex mathematical model is not required. Hence, fuzzy logic could control non-linear system that would be difficult or impossible to model mathematically.

Fuzzy logic is the excellent option when simple mathematical model for the problem is unavailable. Several unique features make it the perfect candidate [41]:

1. it is robust for its tolerance input, controlled output and could design for fail safe easily. Fuzzy logic could perform its functionality as normal even the input information are imprecise and noisy. Its output is a controlled result where the value is always within a defined range regardless of the input variation.
2. fuzzy logic does not require any complex and complicated formula because it is not a mathematical base problem solving methodology. All constrains and system rules that govern the control system are described as a set of rules in natural language which makes the system easier to understand.
3. fuzzy logic is easier to maintain comparing to mathematical control theory. Any number of inputs can be processed and any number of outputs can be generated because of its rule-based inference operation. Implementing any new rules or removing any existing rules from its rule set could be done easily.

Although adding or removing new rules to the rule set could be easily done, the rule set becomes complex, if too many inputs and outputs for a single implementation since rules defining must also define input-output interrelations. Therefore, breaking the control system into smaller parts with multiple fuzzy logic subsystem will be a better decision.

Fuzzy logic is recommended to apply to problems that are either lack of any simple mathematical model or rely on human expert knowledge. Problems that are difficult to formulate mathematical model usually consists of complex and highly nonlinear processes. Problems that rely on human expert knowledge for solution may suffer from making a solution with vagueness and incomplete information. The capability of fuzzy logic in forming an effective and accurate solution for the problems stated above helps to relief the pain using simple mathematical model and linguistic variables. However, if the problems could be easily solved by mathematical model or there exist solutions using mathematical model then mathematical model should be applied.

2.3.1.1 The Origin of Fuzzy Logic

Fuzzy logic is constructed based on the fuzzy theory first introduced by Lofti A. Zadeh in 1965 with his seminal paper “Fuzzy Sets”. Before working on fuzzy theory, Zadeh was a well-respected scholar in control theory. He developed the concept of “state” which forms the basis for modern control theory. In early 60s, he thought that classical theory had put too much emphasis on precision and therefore could not handle the complex system.

Since the birth of fuzzy theory, fuzzy theory has been sparking controversy. Some scholars endorsed the idea and began to work in this field, others objected by claiming that

probability is sufficient to characterize uncertainty and any problems that can be solved by fuzzy theory and can be solved equally well or better with classical theory [41].

Fuzzy theory establishment is largely due to the dedication and outstanding works of Zadeh. Zadeh proposed most mental concepts in fuzzy theory in the late 60s and early 70s. After the introduction of fuzzy sets in 1965, he proposed the concepts of fuzzy algorithms in 1968; fuzzy decision-making in 1970 and fuzzy ordering in 1971. In 1973, he published another seminar paper “outline of a new approach to the analysis of complex system and decision processes” which established the foundation for fuzzy control. In this paper, he introduces the concept of linguistic variables and proposed to use fuzzy IF-THEN rules to formulate human knowledge.

In 1975 Mamdani and Assilian established the basic framework of fuzzy controller and apply the fuzzy controller to control a steam engine. Their results were published in seminar paper in fuzzy theory “An Experiments in linguistic synthesis with a fuzzy logic controller”. The application of fuzzy theory in the fuzzy steam engine controller demonstrated the field was promising. Unfortunately, no major research institute put intensive research onto this field. However, in the early 80s, Japanese engineers found that fuzzy controller is very easy to design and worked very well for many problems. Yasunobu and Miyamoto from Hitachi have successfully developed a fuzzy control system for the Sendai subway which completed its construction in 1987. The success of fuzzy system in Japan surprised the researchers in US and Europe. In 1992, the first IEEE International Conference on Fuzzy Systems was held in San Diego. Fuzzy theory has gained its acceptance finally [26].

2.3.1.2 The Theory of Fuzzy Logic

Set theory is useful as classification and grouping element together and is usually part of the problem solving procedure. In classical set theory, the universe of discourse U contains N possible element in a selected context. A set in the universe of discourse U could be defined by listing all elements in the set or it could be described with its characteristics that must be fully satisfied by all members of the set. If one of the defined characteristics is not satisfied, the element is not considered as a member of the set, which means an element in U could only be identify as in the set or not in the set. If a membership function are applied to each element in U for the purpose of measuring the membership of set A (a set in U with a number of element in it), the membership function shall return value 1 for the element that belongs to set A and return the value 0 for the element that does not belongs to set A [26]. The membership function shall be denoted as

$$\mu_A(x_i) = \{1 \quad x \in A; 0 \quad x \notin A$$

where

μ_A is the membership function,

x_i is the i-th element in the universe of discourse U and

i =1,2,3,…N

Hence, the membership function of classical set can only be either one or zero. However, in realistic, not every classification has clear and sharp boundary. As a result, the classical set theory is not applicable in many cases.

Fuzzy set is the core concepts of fuzzy logic. Fuzzy set defines the membership of an element in U to a set as the degree of membership, a representation of a classification.

Instead of using step function that could only return zero or one, fuzzy set is using a continuous membership function that could return any real number within the range of [0,1]. The range of [0,1] is taken from the classical set theory membership function. Fuzzy set always have a one-to-one correspondence with its membership function. Fuzzy set could be considered as the superset of classical set theory which has been extended not only to handle member of the set (value 1) and non-member of the set (value 0), but it can also handle the partial set membership– the fuzzy membership (value between 0 and 1). This allows large flexibility in inferring information and in modeling problems that are too complex.

The fuzzy set is usually characterized using fuzzy property, a characteristic that does not have clear boundaries. It reduces the strictness of classical set, changing from one classification to another classification with the values changes of 1. For example, age above and equal 30 is OLD and age below 30 is YOUNG, and by having a 30 years old birthday today causes someone to change from YOUNG to OLD in a single day. This is not logic when classical set theory is applied in such example. If using fuzzy set theory, let say age above and equal 30 is OLD and age below and equal 20 is YOUNG, then it is interpreted as someone is gradually changing from YOUNG to OLD in between the age of 20 and below 30. Age 25 could be defined as 0.5 member of OLD and 0.5 member of YOUNG. The fuzzy set is proven more practical and realistic than the classical set theory in solving real life problem.

The following is some of the basic concept of fuzzy set [41]:

1. support of fuzzy set

   the support of a fuzzy set is all the elements in U that have nonzero membership value, $\mu(x) > 0$

2. fuzzy singleton

   fuzzy singleton is a fuzzy set that only consist of 1 element in U that have nonzero membership value.

3. height of fuzzy set

   the height of a fuzzy set is the largest membership value that could be returned from the membership function. For example, value 1 is the height of the fuzzy set that has a membership function with the return value in the range of [0,1]. The fuzzy set with the height of 1 is also known as the normal fuzzy set, a fuzzy set that has been normalized.

4. center of a fuzzy set

   for the fuzzy set with its height which is a finite number, the mean value of all elements in U that have nonzero membership value is the centre of fuzzy set.

5. Union of fuzzy set

   The simplest and common definition of union for fuzzy set A and fuzzy set B in U is $\max[\mu_A(x), \mu_B(x)]$

6. intersection of fuzzy set

   the common and simplest definition of intersection for fuzzy set A and fuzzy set B in U is $\min[\mu_A(x), \mu_B(x)]$

7. complement of fuzzy set

   for normal fuzzy set, the complement of fuzzy set A is defined as $1-\mu_A(x)$

8. equality of fuzzy set

   fuzzy set A and fuzzy set B are considered equal if and only if $\mu_A(x) = \mu_B(x)$ for all $x \in U$.

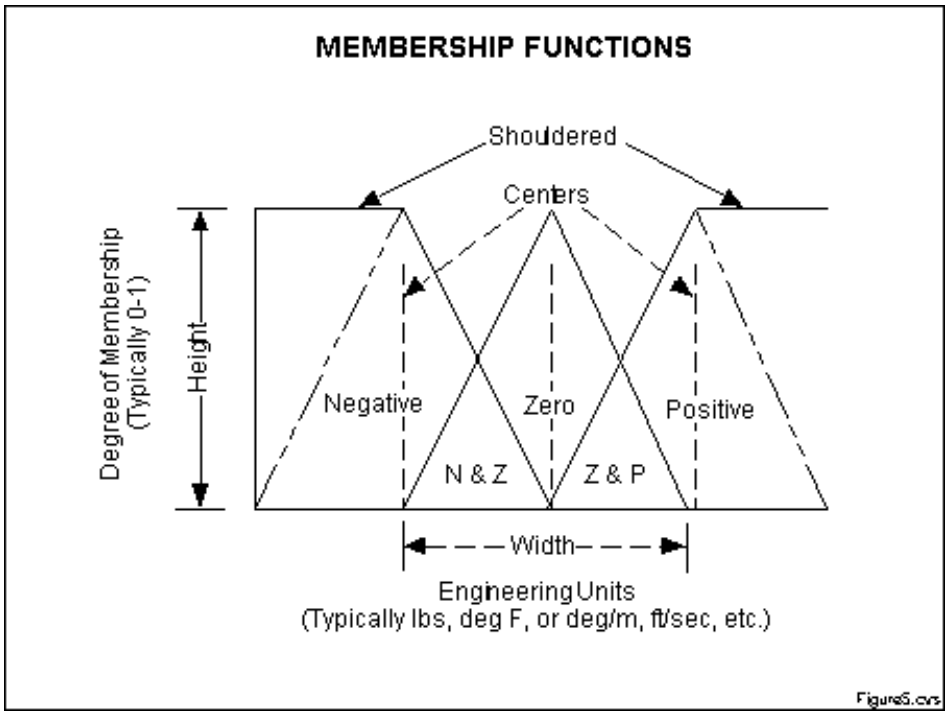

Figure 5: Explanation of membership function

source: [41]

Linguistic variable is another core concept in fuzzy logic that applies fuzzy set theory. It enables the capability to formulate ambiguous description in natural language into precise mathematical terms that could be working with the complex system. Linguistic variable could be considered as the most fundamental element in human knowledge representation. If a variable is taking numerical value as its value, it is called a numeric variable. If a variable is taking words in natural language as its value, it is called a linguistic variable and the words are known as the linguistic value [26].

A linguistic variable is a noun which is used as the name and defines the context of Universe of Discourse for its linguistic value. Each linguistic variable has a number of linguistic values and each linguistic value is characterized by fuzzy set that is defined in universe of discourse. The number of linguistic variable depends on the necessity and

complexity of the system. Every input of the system may be associated with more than one linguistic variable directly or indirectly through some form of computation. The membership function of each fuzzy set will measure the degree of membership for each input. Triangular, trapezoidal and bell (Gaussian function) are some of the common shape of membership function.

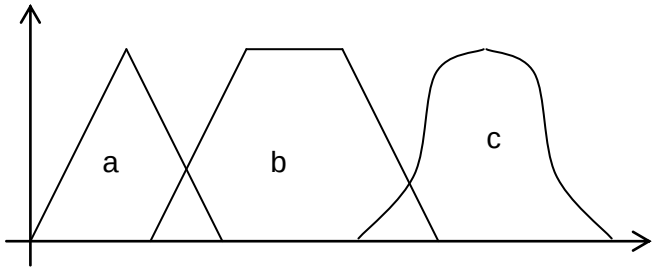

Figure 6: Common shape of membership function
a) triangular, b) trapezoidal and c) bell

The formal definition of linguistic variable is characterized by (X, T, U, M) [26] where

X is the name of linguistic variable, e.g. Buffer length,

T is the set of linguistic value that under this, e.g. {SHORT, MEDIUM, LONG}

U is the actual physical input value that linguistic variable can accept, e.g. [0, 100]

M is the semantic rule that relates each linguistic value T to the actual input value U using the membership function, e.g.

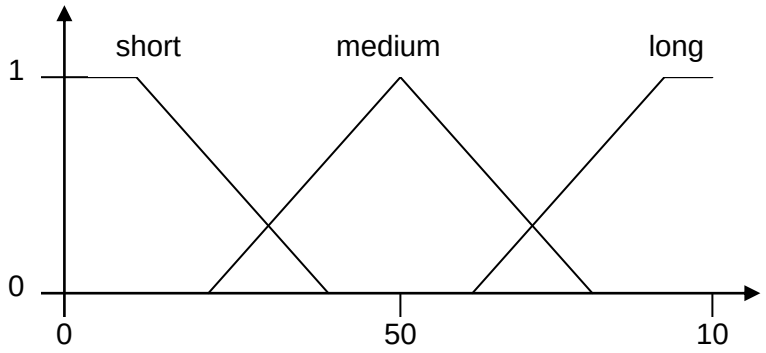

Figure 7: Association of linguistic value with the physical value of buffer length using membership function

After determining the linguistic variable, fuzzy rule can be defined from human expert knowledge. Fuzzy rule is an IF-THEN statement in the form of IF A THEN B where antecedent A and consequent B are fuzzy set. The antecedent describes the condition that must be fulfilled by the current system state. If the condition of the rule is met by the current system state then the rule is a match. The consequent of the fired rule then assigns the linguistic value to the output variable. For example, IF BUFFER_QUEUE is LONG THEN COST is HIGH. A single statement or a compound statement could form the antecedent and consequent of a fuzzy rule. The compound statement is a composition of single statement using the connections “and”, “or” and “not” which represent fuzzy intersection, fuzzy union and fuzzy complement respectively. The following is an example of compound statement, BUFFER_QUEUE is LONG and BUFFER_CHANGING_RATE is HIGH.

The collection of fuzzy rule forms the heart of a fuzzy logic system which is known as the fuzzy rule base. The fuzzy rule base is the transformation of human knowledge into the fuzzy logic system. The completeness of fuzzy rule base is determined by having at least one rule fires for any input value. Incomplete fuzzy rule base may have some unexpected decision from the fuzzy logic system that will cause the entire system malfunction. A fuzzy

rule base is consistent if there are no rules with the same antecedent but different consequent. However consistency is not a critical factor, as the fuzzy inference engine will automatically average the conflicting rules to produce a compromised result.

Fuzzy logic system is also known as fuzzy inference system for its rules inference process. In general, a fuzzy inference system involves the following four steps in sequence [41]:

1. Fuzzification. Fuzzification is the process to fuzzify the numeric value of selected input variables into linguistic value of the corresponding linguistic variables. Fuzzy inference system fuzzifies numeric value by plugging the selected input variable into horizontal axis and projecting vertically to the upper boundary of the membership function. Each input value may transform into multiple linguistic values for each corresponding linguistic variable, such as 0.5 of LONG and 0.5 of MEDIUM for BUFFER_QUEUE.
2. Rule Inference. Rule inference is the process of evaluating the truth value for all human expert defined rule. The consequence of each rule that is fired (having its antecedent matching the current system state) will produce an output linguistic value. Many inference methods exist to perform rule inference. The most common inference methods are MAX-MIN inference and MAX-PRODUCT inference. If using MAX-MIN inference method, fuzzy union uses the maximum and minimum operation for fuzzy intersection. If using MAX-PRODUCT inference method, fuzzy union uses the maximum operation and uses multiplication for fuzzy intersection. The selection of the inference method will determine the output value. The maximum membership value of all linguistic values (in the antecedent of a fired rule) is the membership value for the output linguistic value (in the consequent of a fired rule) if all linguistic values are related by fuzzy union. If all linguistic values (in the antecedent of a fired rule) are related by fuzzy intersection, then the minimum or product membership value of linguistic values will be the membership value for the output linguistic value (in the

consequent of a fired rule). For example, the rule is IF A is LONG and B is FAST then C is REDUCE while A is 0.5 of LONG and B is 0.6 of FAST. If using MAX-MIN inference, the output is 0.5 of REDUCE and if using MAX-PRODUCT inference, the output is 0.3 of REDUCE.

3. Composition. During the rule inference process, each fired rule will produce an output value. As a result, there exist a number of output values for a linguistic variable. The composition process is to combine all output values of a linguistic value to become a composite output value of a linguistic value. The common composition methods are MAX or SUM. If using MAX composition, the composite output value is calculated by taking the maximum membership value of all output values. If using SUM composition, the composite output value is calculated by taking the sum membership value of all output values. The composition process may produce membership value for more than one linguistic value as result of an output variable. For example, the output result is 0.75 of REDUCE and 0.3 of ZERO and 0.1 of INCREASE for linguistic variable SPEED.
4. Defuzzification. Defuzzification is the process of reverse fuzzification that converts the linguistic value into a numerical value. The defuzzification process is necessary as the system inputs are accepting numerical value only. There are many defuzzification methods for selection. The common defuzzification methods are centroid of area, mean of maximum, smallest of maximum and largest of maximum. The following is the formula for the centroid of area defuzzification method:

$$y = \frac{\sum_{i=1}^{M} \mu_i w_i}{\sum_{i=1}^{M} w_i}$$

where y is the defuzzification value,

u_i is the membership value for each linguistic variable and

w_i is the x-axis (horizontal) value for the linguistic variable.

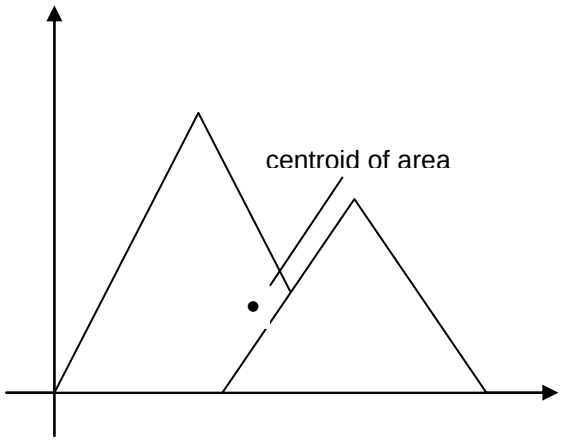

Figure 8: Centroid of Area Defuzzification Method

Mean of maximum is the mean of the maximum membership value. Smallest of maximum is the smallest horizontal axis value of the maximum (highest) membership value and largest of maximum would mean the largest horizontal axis value of the maximum membership value.

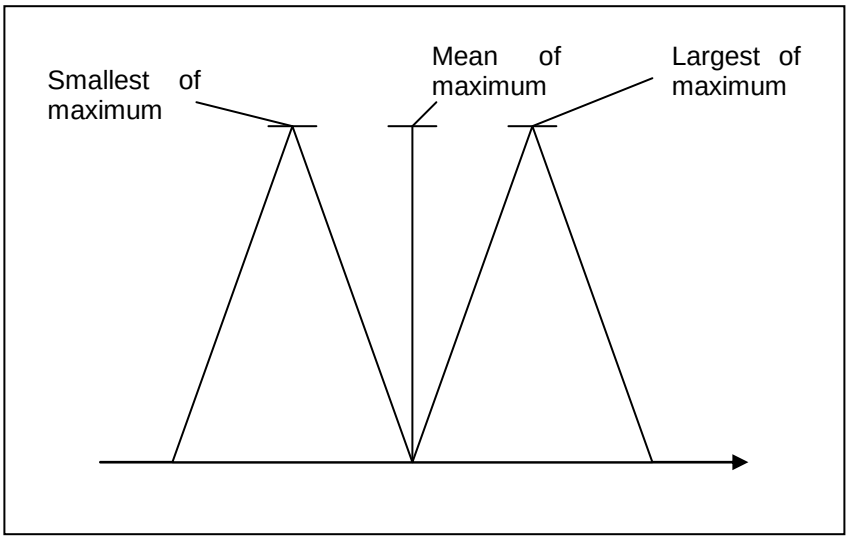

Figure 9: Mean of Maximum, Smallest of Maximum and Largest of Maximum Defuzzification methods

Although it is possible to create multi-input-multi-output fuzzy inference system, building a system in this manner would create much complexity. As the number of input or output increases, the number of fuzzy rules will increase exponentially hence making the system difficult to maintain or control and bound to suffer the curse of dimensionality.

Therefore it is a better practice to create a system of multi-input-single-output. If there's a need to create multi-output, it is advisable to break the system down to sub-system in order to achieve the multi-output result. This way, the system is easier to maintain because the system does not suffer from the curse of dimensionality.

Fuzzy system has been applied to a wide variety of field ranging from control, signal processing, communications, integrated circuit, expert system, and medicine.

2.3.1.3 Fuzzy Control Applications

Varies traffic characteristics and the need for accurate judgment in ATM networks is causing the use of traditional control methods to be difficult. Fuzzy logic are applied to solve the controlling problem to incorporate expert knowledge and therefore allow interpretation of control actions. The following are some implementations of fuzzy logic in ATM.

Fuzzy logic Based ATM Policing [25] is an application of fuzzy logic as an ATM policer controller. In *Fuzzy logic Based ATM Policing* [25], the ratio of the measured mean burst length of the incoming traffic to the negotiated mean burst length and the ratio of the measured mean rate to the negotiated mean rate are used as input variables.

In *Local Area Network Traffic Characteristics* [17], a thorough study is done with real traffic to prove that heuristic determination of the congestion can be expressed very accurately by using fuzzy rules. The result is supported by *Design of a Fuzzy Traffic Controller for ATM Networks* [33], *Fuzzy Backward Congestion Notification (FBCN) Congestion Control in Asynchronous Transfer Mode (ATM) Networks* [4] and *Self-tuning Fuzzy Traffic Rate Control for ATM networks* [32]. In *Design of a Fuzzy Traffic Controller for ATM Networks* [33], a global fuzzy traffic controller that handles both admission control and congestion control via separate mechanisms is presented. The queue's state, queue dynamics (q and Δq) and the measured cell loss are used as inputs in fuzzy congestion controller. The fuzzy congestion controller responds negative control action for congested situation and positive control action for congestion-free situation.

Fuzzy Backward Congestion Notification (FBCN) Congestion Control in Asynchronous Transfer Mode (ATM) Networks [4] presents a fuzzy explicit rate indication BCN controller

that is applicable to available bit rate (ABR) traffic source. The fuzzy controller is fed with the buffer value and the buffer changing rate. The controller responds with new normalized explicit rate to the source switch.

In *Self-tuning Fuzzy Traffic Rate Control for ATM networks* [32] the network congestion level is monitored through the buffer length, its dynamics, the measured cell losses, and the ABR transmission demand. A feedback signal is calculated to throttle the source rate.

In *Routing in Multimetric Networks using a Fuzzy Link Cost* [5], fuzzy logic is introduced as a tool to solve the complicated multimetric routing problem. The multimetric routing mechanism becomes complicated as each link is described in terms of multiple metrics and routing path selection depends on the request requirements. Some metrics have negative effect on the routing path selection while others have positive effect on it. The routing path selection becomes more complicated when the algorithm tries to assign a routing path that just satisfies the request requirements rather than assigning it to a path that satisfies the requirements more than it requested.

The proposed fuzzy logic system in [5] defines fuzzy cost for each link based on the available crisp value metrics. The fuzzy link cost is determined based on the 2 selected metrics: estimated end-to-end delay and available bandwidth. Estimated end-to-end delay is used to guarantee that the real time data are delivered with minimum delay and available bandwidth is used to determine the acceptance of the request.

In experiment of [5], the proposed fuzzy link cost routing is compared with a crisp routing algorithm that adopting both shortest path and stochastic routing. The experiment result shows the fuzzy link cost has higher throughput compared to the crisp routing. The higher throughput is due to the proposed fuzzy link cost always directing request with smaller

bandwidth requirement to links that almost fit their requirements to save the links with larger bandwidth for video transmission. The fuzzy link cost routing always has higher average utilization as it has higher throughput and efficiently utilize low bandwidth links for low requirements request. However, crisp routing has lower delay during transmission because it has lower overhead and always assigns calls to the path that allocates resources more than required.

2.3.2 Neural network

Neural network is inspired by neuropsychological knowledge trying to imitate the human intelligent. The human intelligence is sourced from the human brain. In order to imitate human intelligence, it is necessary to understand the human brain structure and its behavior. This understanding is the study of artificial neural network.

The human brain is the central human nervous system and it can be represented by neural network. The neural network receives the stimuli information from the receptors and generates responses to the effectors. The nerve cells of the nervous system are called neurons which specialize in carrying messages through electrochemical process. Each neuron has dendrite which acts as receptive zones to bring information to the cell body and uses axon as the transmission line to carry information away from the cell body. The cell body which is known as soma is the triggering zone for a neuron. The sum is responsible to create a brief voltage pulse (a sudden voltage increase for about 1ms) that will propagate through the axon. Axon of each neuron has numerous branching points forming an axonal tree links to other neurons. The interaction between two neurons relies on synapse. A synapse consists presynaptic ending that contains neurotransmitter and postsynaptic ending that contains receptor sites [36].

The majority of neurons encode their outputs as a series of brief voltage pulses. These pulses, commonly known as action potentials or spikes, originate from cell body of neuron and then propagate across the individual neuron at constant velocity and amplitude. The spikes propagate along lengthy axon that extends from the cell body. The axon uses the nodes of Ranvier as a repeater to restore the shape of propagated spikes. When reaching the branching point of axonal tree, spikes are duplicated to enter each branch of axonal tree. In this way, a spike from a neuron can be transmitted to many other neurons. However, for the purpose to transmit the spike to another neuron, the spikes need to pass through synapse.

When a spike enters a synapse and reaches the presynaptic ending, it will trigger the migration of vesicles filled with neurotransmitters toward presynaptic membrane. The vesicle membrane will fuse with the presynaptic membrane releasing the neurotransmitters into extracellular fluid. Whenever a neurotransmitter molecule binds to a receptor on the postsynaptic ending of the synapse, it will cause changes to membrane voltage by a few millivolt. These potential changes become EPSPs (excitatory postsynaptic potentials) if they increase the postsynaptic potential and become IPSPs (inhibitory postsynaptic potentials) if they decrease the postsynaptic potential. The postsynaptic potentials created by all synapses converging on a single neuron are transmitted by the input tree (dendrite tree) to the trigger zone at the cell body of a neuron. Whenever the sum of the continuously arriving postsynaptic potential at the trigger zone reaches the firing threshold, the neuron will fire an action potential (spike) through its axon [48].

A spike always looks alike but it is different from the postsynaptic potentials. The postsynaptic potentials depend on the particular synapse that it is in. The size of

postsynaptic potentials relies on the pattern of spikes that have reached the synapse in the past and depends on the interaction of the spikes with the firing activity of the postsynaptic neuron. Due to the uncertainty of the size of postsynaptic potentials, the firing time of the postsyanptic neuron becomes variable. In certain extreme occasions, the postsynaptic neuron will not fire any spike at all for the reason of insufficient membrane potentials. Besides the reason of insufficient membrane potentials, a neuron will not fire any spike within its absolute refractory period regardless of how large its membrane potential. The absolute refractory period is a few milliseconds right after a neuron has fired. Then for a few further milliseconds the neuron is still reluctant to fire which requires larger membrane potentials than usual as the neuron is in relative refractory period [48].

General artificial neural network model that has network size of s indexes its neurons from 1 to s where s is the number of neuron in the network. Some of the neurons in the neural network may function as the interface to the external inputs or outputs. Assuming the neural network is having n input neurons and m output neurons with the remaining neurons in the network as hidden neurons. Note that, hidden neurons may not exist for certain neural network model. Each neuron is densely connected to other neurons as a directed graph demonstrating the neural network architecture whereby each connection from neuron i to neuron j is associated with a synaptic weight, w_{ij}. For the pair of neurons that do not have connection between them are considered as having zero synaptic weight. [38]

Every artificial neural network will undergo training or a learning process which will continuously update the state of neurons in the network or the synaptic weights. The neurons state or synaptic weights will be updated according to defined change function which gradually brings the neural network to a desired state. At the beginning, the neural

network is normally initialized with practical random number or may be derived from external inputs. A network state is then updated by a subset of neurons which recalculate their current state using other neurons state and the corresponding synaptic weights. Finally, a global output from the network is read as a solution. [38]

In the survey conducted in *Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research* [31], a number of neural network models are implemented to solve the combinatorial optimization problem since the first use of Hopfield network in solving it. According to the survey, generally the neural network models that could solve the combinatorial optimization problem are under two main approaches: Hopfield model approach and self-organizing model approach. Among all combinatorial optimization problem, Traveling Salesman Problem is the most popular to be used to measure the efficiency and effectiveness of the neural network.

2.3.2.1 Travelling Salesman Problem

Traveling Salesman Problem is that a salesman needs to travel through N selected cities in the shortest route with the following explicit and implicit conditions:

1. Every city is visited only once in the tour.
2. Every selected city must be visited in the tour.
3. Each time the salesman could only visit a city.

2.3.2.2 Hopfield Network Model

In the year 1982, John Hopfield introduced a neural network model that has content-addressable memory (CAM) capability. The neural network was named after him as Hopfield network. A CAM system is the capability for the system to retrieve a stored item when the system has incomplete but sufficient information. Then in year 1985, John Hopfield and David Tank have successfully demonstrated solving Traveling Salesman Problem using modified Hopfield network. Since the use of TSP by Hopfield and Tank, TSP has become the popular reference example for solving combinatorial optimization problem.

Hopfield network is an unsupervised, recurrent neural network as the neural network is trained without the knowledge of final desired state and the output of each neuron is self-feed at the next iteration. The principal of Hopfield network is to minimize the network energy that calculates through the energy function which could lead the Hopfield network into a stable state whereby the stable state is the solution. In the application of Hopfield network on TSP that was proposed by Hopfield and Tank, the network energy function is considered as the objective function. The objective function will be minimized during the learning iteration. The objective function includes the constraints of the problem as penalty terms.

For the TSP problem with n cities, the Hopfield network is constructed as NxN network where each column represents visiting city and row represent the sequence of tour. The Hopfield network is having symmetric weights with zero diagonal elements which is $w_{ij}=w_{ji}$

and w_{ii}=0. Hopfield has proved that if synaptic matrix is symmetric with zero diagonal elements then the energy function is decreased by any update as shown below:

$$E = -\frac{1}{2}\sum_{ij} v_i w_{ij} w_j - \sum_i b_i v_i$$

The Hopfield network is a mesh connected recurrent network where each neuron output is connected to every other neuron in the network including its own. The Hopfield network uses perceptron as its neuron. A perceptron is a neuron that uses summation of all connected synapse from other neuron multiplies with the synaptic weight and its bias to calculate its input state. Hence, the neuron input state u_{xi} is denoted by

$$u_{xi} = \sum_{y=1}^{n}\sum_{j=1}^{n} v_{yi} w_{xi,yj} + b_{xi}$$

where n is the number of neuron,

xi and yj is the index of neuron,

u_{xi} is the input state of neuron at x row i column,

v_{yj} is the output state of neuron at y row j column,

$w_{xi,yj}$ is the synaptic weight between neuron at x row i column and neuron at y row j column and

b_{xi} is bias of neuron at x row i column.

The neuron will then use the input state to calculate its output state v_{xi} using a sigmoidal function. The output function selected here is as following:

$$v_{xi} = \frac{1}{2} \times 1 + \tanh\left(u_{xi}\right)$$

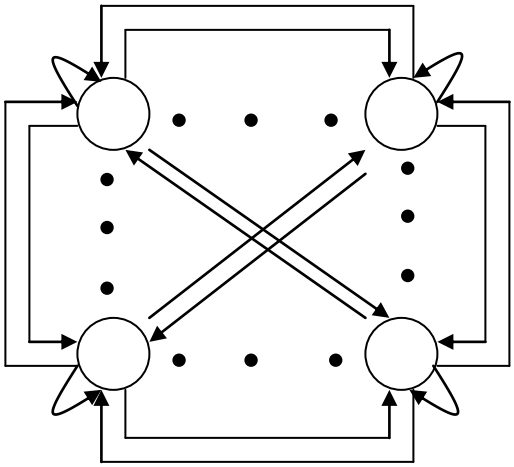

Figure 10: Hopfield network model

The objective function that is used by Hopfield and Tank to solve the TSP is

$$E = \frac{A}{2}\sum_{x=1}^{n}\sum_{i=1}^{n}\sum_{j\neq i}^{n} v_{xi}v_{xj} + \frac{B}{2}\sum_{i=1}^{n}\sum_{x=1}^{n}\sum_{y=1,\,y\neq x}^{n} v_{xi}v_{yi}$$
$$+ \frac{C}{2}\left(\sum_{x=1}^{n}\sum_{i=1}^{n} v_{xi} - n\right)^2 + \frac{D}{2}\sum_{x=1}^{n}\sum_{y=1}^{n}\sum_{y\neq x,\,i=1}^{n} d_{xy}v_{xi}\left(v_{y,i+1} + v_{y,i-1}\right)$$

where A, B, C, D is coefficient factor,

n is number of city,

v_{xi} is output state of neuron at x row and i column and

d_{xy} is distance between city x and y.

The first term in the energy function ensures that each row only has a value of 1 as the salesman cannot visit the same city more than once. The second term ensures that each column only has a value of 1 as the salesman could not be at two different cities at the same time. The third term enforces that all cities are visited without missing any of them. The fourth term calculates the total length of the tour to find the shortest route for the tour.

From the defined objective function, there are two methods to identify the synaptic weights matrix and bias for each neuron. The first method is to rewrite the objective function in the following form

$$E = -\frac{1}{2}\sum_{ij} v_i w_{ij} w_j - \sum_i b_i v_i$$

The second method is to determine the local function of motion u_{xi} from the objective function where the objective function should always decrease.

$$\frac{dE}{dt} = \sum_i \frac{\partial E}{\partial v_i} \cdot \frac{dv_i}{du_i} \cdot \frac{du_i}{dt}$$

if $u_{xi} = -\frac{\partial E}{\partial v_i}$, then $\frac{dE}{dt} = \sum_i \frac{dv_{xi}}{du_{xi}} \cdot \left(\frac{\partial E}{\partial v_{xi}} \right)^2 \le 0$

Since

$$\begin{aligned} u_{xi} &= -\frac{\partial E}{\partial v_i} \\ &= -A\sum_{j \neq i} v_{xj} - B\sum_{y \neq x} v_{yi} - C\left(\sum_y \sum_j v_{xj} - n \right) - D\sum_{y \neq x} d_{xy}\left(v_{y,i+1} + v_{y,i-1} \right) \end{aligned}$$

when rewrite as

$$u_{xi} = \sum_{y=1}^{n} \sum_{j=1}^{n} v_{yj} w_{xi,yj} + b_{xi}$$

the synaptic weights and bias are identified as

$$w_{xi,yj} = -A\delta_{xy}\left(1 - \delta_{ij}\right) - B\delta_{ij}\left(1 - \delta_{xy}\right) - C - Dd_{xy}\left(\delta_{j,i+1} + \delta_{j,i-1}\right)$$

and $b_{xi} = Cn$

where $\delta_{xy} = \begin{cases} 1 & if\ x = y \\ 0 & if\ x \neq y \end{cases}$

The procedure for Hopfield learning process is:

1. Initialize every neuron with random number between 0 and 1.

2. Randomly select a neuron and recalculate its input state.
3. Using the new input state, recalculate its output state.
4. Repeat step 2 to 3 until all neurons are updated to complete iteration. A neuron will not be updated more than once in an iteration.
5. Repeat the iterations from step 2 to 4 until the network become stable where the network achieves its minimum energy level.

2.3.2.3 Kohonen Self-Organizing Neural Model

The self-organizing network is a type of unsupervised neural networks that organizes the structure or neuron state to form clusters of ordered data based on the input data patterns and features. Among all the self-organizing neural networks, Kohonen self-organizing feature map introduced in year 1982 is the most popular reference neural model. The Kohonen neural model transforms the input data into a one- or two-dimensional array of neurons in a topologically ordered manner. The Kohonen neural network achieves its learning through the adaptation of the weights connecting the input source to the selected output neurons. Output neurons weight adaptations drives the neurons to order themselves to the data clusters based on the presented input data.

In the application on the Traveling Salesman Problem, the output neurons are organized as a ring, the best suitable structure that matches the requirement of TSP. The ring structure guarantees that the starting point of the route will be the ending point after visiting all cities regardless of which city it begins. The Kohonen neural model by referring [36] and [44] for TSP problem consists of two layers only, the input layer and the output layer. The output layer is having m neurons in a ring where m is equal or more than the number of cities. The input layer is having two input node. Since all cities are mapped on

Cartesian coordinates, the input data are the Cartesian coordinate of the city. Each input node is connected to every output neuron with the weight wij, for the connection from input node i to neuron j. The input data is presented to all neuron simultaneously.

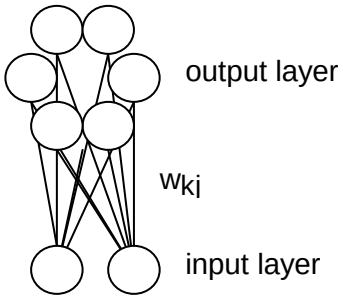

Figure 11: Ring self-organizing neural network

Whenever a vector of input data is presented to the neural network, only a few selected neurons are going through the weight adaptation. In order to identify the selected neurons, a competitive process takes place and will only have a winner in the competitive process. The "closest" neuron (according to some definition, in this case will be the Euclidean distance) to the input vector is the winning neuron. If using the neuron index, ind(x), to identify the winner, the competitive function finds the minimum distance denoted as

$$ind(x) = \min \sqrt{(x_1 - w_{1j})^2 + (x_2 - w_{2j})^2}$$

where x is the input vector for all neuron j in the output layer.

The selected winning neuron tend to excite the neurons in its topological neighborhood where the winning neuron is located at the center of the topological neighborhood. The excited neurons will then go through the adaptation process together with the winning neuron. The topological neighborhood is not static across the entire learning process and not every neuron is having the same adaptation effect. The neurons which are "nearer" (in the sense of lateral distance) to the winning neuron will have larger weight changes compared to the neurons which are further away. The topological neighborhood will

decrease constantly while shrinking down its size during the iteration. Let $h_{i,j}$ denote the topological neighborhood having the neuron i at the center and encompassing a set of excited neuron j with the lateral distance between neuron i and neuron j denoted by d_{ij}. The typical choice for the neighborhood function is the Gaussian function

$$h_{i,j}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right)$$

where $\sigma(n)$ is the threshold width that determines the size of the topological neighborhood. The lateral distance, $d_{i,j}$, in this case would be the Euclidean distance between the winning neuron and neuron j by considering the neuron weight as the location in Cartesian plane.

$$d_{i,j}^2 = (w_{1i} - w_{1j})^2 + (w_{2i} - w_{2j})^2$$

The size of the topological neighborhood that is shrinking with time is defined by

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right)$$

where n is the number of iteration, σ_0 is the initial value and τ_1 is the time constant.

Having the set of chosen neurons for adaptation process, their synaptic weights need to be updated. The adopted learning function is defined as following

$$\Delta w_{kj} = \eta h_{i,j}(x_{kj} - w_{kj})$$

where k is the index of input nodes, η is the learning rate. Hence, the weight for neuron j at time n+1 shall be

$$w_{kj}(n+1) = w_{kj}(n) + \eta(n) h_{i,j}(n)(x_{kj} - w_{kj}(n))$$

The chosen learning rate is gradually decreasing with time based on the heuristic decision. The learning rate is as follows:

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right)$$

where τ_2 is a time constant.

The Kohonen model algorithm in summary is as follows:

1. Initialization. Initialize the synaptic weights which could also be considered as the Cartesian location of neuron. Three initialization methods have been introduced to produce a more efficient Kohonen model. The first method is randomly distributing the neurons within the input data space. The second method is to distribute the neurons at the center of the map and the third method is to distribute the neurons at the outer ring that is having all cities within the ring. For simplicity, the second method was adopted.
2. Competition. In the competition process, the input data is presented to all neurons and a winning neuron that is "closest" to the input data will be selected. The winning neuron is determined through the function

$$ind(x) = \min \sqrt{(x_1 - w_{1j})^2 + (x_2 - w_{2j})^2}$$

3. Cooperation. In the cooperation process, the neurons within the topological neighborhood of winning neuron are selected for adaptation process. The topological neighborhood is determined using

$$h_{i,j}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right)$$

4. Adaptation. The adaptation process is the process of updating the synaptic weight for the excited neurons that is located in the neighborhood. The update function is

$$w_{kj}(n+1) = w_{kj}(n) + \eta(n) h_{i,j}(n)\left(x_{kj} - w_{kj}(n)\right)$$

5. Repetition. After each excited neuron passes through the weight updating, the process is continued to step 2 until yielding a satisfactory result.

2.3.2.4 Spiking Neural Network Model

2.3.2.4.1 Revisit on biological neural

As written in the previous section, biological neurons transmit information by encoding it as a series of spikes while every spike has the same size and looks alike. Since all spikes are the same, the form of spikes does not carry any information. Every spike is usually well separated by neuron's refractoriness. A neuron will not fires a second spike immediately after the first spike regardless of the input strength. The minimum distance between two spikes from a neuron is equilibrium to neuron absolute refractory period where a neuron will not fires a spike during absolute refractory period. The absolute refractory period is then followed by the relative refractory period where a neuron faces difficulty in firing a spike which variably separates the spikes.

Since the form of spike does not carry any information, the possible information encoding schemes is by using the number or the timing of spikes. Information encoding using the number of spikes is generally known as the rate codes which are commonly taking the average number of spikes over a defined period. If using the timing of spikes as the information encoding scheme, the time differences between spikes are the spikes codes.

The following are the various encoding scheme based on rate codes and spikes codes [52]:

A. Rate codes

1. Rate as spike count. This is the most commonly used definition of rate code which refers to the number of spikes in a defined time window. In order to get reasonable rates, several spikes should fire within the time window. This rate code works well only in the case where the input is constant or varies slowly and does not require a fast reaction. If non-constant input is presented to this rate code, it may have faulty information interpretation. The rate code slow detection and response is due to it requiring at least two time window for an action. Hence, this rate code is not likely the encoding scheme adopted by the biological neurons.
2. Rate as a spike density. A neuron is stimulated with some input sequence and the same stimulation sequence is repeated several times to record the neuron response. The recorded neuron response is then used to measure the neuron activity within the stimulation interval length. The neuron activity is calculated by summing the number of spikes of all repetition divided by the number of k repetition. The calculated neuron activity is then further divided by the stimulation interval length to get the spike density. The spike density is measured in the unit of Hz and often known as the firing rate of the neuron. The problem with this rate code is that it expects to repeat this scenario every time. However, this rate code is practical if there are large populations of independent neurons that receive the same stimulus.
3. Rate as population activity. Considering a population of neurons having identical properties with the same pattern of input and output connections, the spikes in the population j are sent to another population k. Each neuron in population j receives input from all neurons in population k. The population activity is then calculated by summing the number of spikes over all neurons in the population j divided by the number of neurons in population j and further divides by Δt a small time interval. The result of calculation is the average population rate. The population activity may

vary rapidly and reflect the changes nearly instantaneously. However, it requires homogeneous population of neurons with identical connection which is not realistic.

B. Spike codes

1. Time-To-First-Spike. This spike code uses the timing of the first spike after the reference signal for each neuron as the information encoding scheme. A reference signal in this spike code is the internal signal of an event occurrence. The timing of first spike is believed to have all the information about the new stimulus because a neuron fired earlier signals a strong stimulation while firing later signals a weaker stimulation. In order to implement the ideal model of this encoding scheme, each neuron is shut off by inhibition after the neuron has fired a spike until the beginning of next stimulus. Neurons are ready to fire its next spike immediately after the inhibition. Since the first spike contains most of the information, the reaction responses are faster.
2. Phase. The Time-To-First-Spike can be applied in the scenario where the reference signal is not an event, but a cyclical signal. The spike coding encodes information in the cyclical internal reference signal phase. If the input does not change, then the same pattern of phases repeats.
3. Correlations and synchrony. Spikes from other neurons could be used as the reference signal for a spike code. Synchronized firing between neurons could have signify special events and convey more information. The synchronization firing of neurons which is similar to the idea of self-organizing maps indicates that the neurons are correlated and belongs to the same group. In general, neurons synchronization firing and the precise timing could hold more meaningful information.
4. Stimulus reconstruction and reverse correlation. This coding scheme tries to reconstruct the complete time course of the stimulus using analysis from reverse

correlation. Reverse correlation calculates the average time course of each stimulus with identical response by averaging the time course in a time window with a defined time interval immediately before the spike firing for repetitive runs on the set of stimulus. Hence, reverse correlation is also known as spike-triggered average. Results from reverse correlation analysis suggest that each spike signifies the time course of the stimulus before the spike. If this is correct, a reconstruction of the complete time course of the stimulus should be possible.

2.3.2.4.2 Spiking Neuron

Spiking neurons referred here is equivalent to the spiking neural network. Hereafter, these terms are interchangeable in this writing. Spiking neuron is differing from the common neural network neuron, such as McCulloch-Pitts neurons and sigmoidal gates in the vital aspect of computation. Spiking neurons network is more biologically plausible compared to the common neural network models. It uses the same mechanism as the biological neural system. If using the analysis of the potential encoding scheme adopted by the biological neurons, the common neural network neurons (with sigmoidal gate) will use rate codes because the neurons output is a real number. The spiking neuron network adopts spikes codes instead of rate codes because spikes code is the preferred encoding scheme adopted in biological neurons.

Maas has proven that spiking neurons could compute any given function faster than common neural network model, regardless of how complex the function is in his analysis on spiking neurons computation capability. The detail of spiking neural network that uses modified Time-To-First-Spike is described in the following chapter.

2.3.2.5 Neural Network Control Applications

The use of NN technique in ATM control falls into four general categories: NN-based admission control, NN-based congestion control and policing, NN-based traffic characterization and prediction and NN-based switch control.

In the first two categories, the prediction and classification abilities of NN are used to either predict and classify incoming traffic, or combine prediction of the expected system performance with congestion adaptation mechanism. For such problems feedforward neural network is the common choice.

NN used in admission control performs classification of acceptable and unacceptable traffic types. NN used in congestion control needs to predict the rate of arrival first so that they can suggest optimal control actions. In the last category, NN is used as generic optimizers and thus Hopfield-type neural network are suggested. The following are examples of studies in applying NN in ATM.

A back propagation neural network is used in [1] and [3] to characterize the statistical variations of the packet arrival process and in *Traffic Prediction using Neural Networks* [13] for traffic prediction. This neural network is capable of capturing the unknown complex relation between the past and future values of traffic. In *A Neural Network Approach for Congestion Control in Packet Switch OBP, Satellite* [19] NN is used for load prediction based on queue status and estimated arrival rate over number of estimated assigned slots.

A window-based scheduling controller is proposed on *Omega Network-based ATM Switch with Neural Network Controlled Bypass Queueing and Multiplexing* [54] using NN for nonblocking ATM switches. A Hopfield type neural network is used to optimize an energy

function that includes a cost for blocking and in-sequence transmission. A set of nonblocking cells is thus transmitted for every time slot. The size of this Hopfield neural network is equal to the product of the number of input buffers times the window size.

In *A Neural Network Implementation of an Input Access Scheme in a High-Speed Packet Switch* [28], an input access scheme using Hopfield neural network for a switch that maintains a separate input queue for each output is presented. The switching fabric is nonblocking. This neural network helps to solve an intractable optimization problem that provides the best scheduling parameters in each time slot.

An adaptive rate-based feedback controller presented in *Congestion Control Mechanism for ATM Networks using Neural Networks* [2] uses a neural network. The neural network uses the number of cells in a multiplexed buffer as a measure of congestion and then feedback the optimal rate to the source traffic. The cost function combines the multiplexer buffer overflow and the level of traffic coding rate for optimal rate calculation. The simulation indicates that NN can be used effectively to reduce the cell loss rate while at the same time they can keep the quality of the transmitted signal virtually unaffected.

Besides controlling solution, neural network is applicable on solving multiple constraints shortest path problem. The author of [23] proposes a recurrent neural network modified from deterministic annealing neural network in [24] and [22] for shortest path problem. The shortest path problem is a combinatorial optimization problem, which minimizes both the number of hops and total associated costs. The proposed recurrent neural network has the ability to process positive and negative cost coefficient (positive is loss and negative is gain) to find the shortest path.

In *Solving Graph Algorithms with Networks of Spiking Neurons* [37], it reveals the ability of spiking neurons to solve shortest path problem. Neurons in spiking neuron network represent the network nodes, while synapses (connection between the neurons) represent links between network nodes. Each synapse is associated with a weight. During learning cycles, the weights are updated by temporal correlation function based on the arrival time of the "spike" signal. The "spike" signal is a signal that propagates throughout the entire network. At the end of the learning cycles, the result of spiking neuron network is the shortest path network.

The paper *A New Neuro-Fuzzy system for Efficient ATM Traffic Control* [10] proposes a neuro-fuzzy system, FasArt, for CAC and UPC problem in ATM traffic control. FasArt is a fuzzy logic system that adopts ART-based (Adaptive Resonancy Theory) neural architectures. ART-based neural network is adopted for its learning and adaptive ability. The proposed FasArt demonstrates low cell loss ratio for a fine usage of link in CAC and better selectivity with low probability of false alarm in UPC.

The solution proposed in [10] inspires the possibility of integrating neural network and fuzzy logic to solve the multiple objectives shortest path problem in ATM. Spiking neurons network, titled "the third generation neural network", and fuzzy link cost system for cost computation are the chosen candidate. Fuzzy logic is responsible for computing each link cost and spiking neuron network is responsible to find the shortest path with minimum hop length and associated cost.

2.4 Conclusion

ATM network is one of the high speed networks which delivers its services with a guaranteed QoS. A pair of VPI/VCI uniquely identifies each connection between source and destination node. In order to ensure the guaranteed QoS, various control such as CAC and UPC are implemented and network information gathering mechanism such as PNNI is applied.

In any case disregard of how powerful the network can be, the network physical resources is always limited while the request of connection is infinity. Using alternative path is able to accept more connections while maintaining the network throughput is inevitable.

Dynamic routing in ATM network is an instance of the NP-complete problems as dynamic routing maximizes its multiple objectives and minimizes its multiple constraints. Any additional objective or constraint will cause the problem complexity increases exponentially.

Various concepts on how to use network resources more cost-effectively are proposed. Although ATM Forum has proposed PNNI routing scheme to assist in ATM dynamic routing, source routing methods seems to burden the border node with intensive computation. PNNI source routing potentially has higher tendency to have crankback problem due to using an out-dated information or aggregated information. The current main stream of dynamic routing is load balancing (distribute the network load evenly across the network) and bandwidth packing (concentrate on maximizing selected resources to reserve resources for request with higher requirement). Both seem to be contrary concepts, have its strength and weakness in different situation whereby they are

potentially able to cover each other's weakness. Modified least load routing in [20] illustrates the incorporation of load balancing and grouping of traffic classes successfully, in order to enhance the overall performance.

In *Dynamic Routing Algorithms in ATM Networks* [15] and *A Mana*gement *System for Load Balancing through Adaptive Routing in Multi-Service ATM networks* [16], the idea of utilizing management place or central node in routing opens a new solution space. By having the efficient routing no longer need to rely totally on the switches themselves. Management plane or central node with up-to-date information reported from each node will oversee the entire network. Having the central node to monitor the entire network performance and an effective routing plan, it relieves border switches from intensive calculation and speed up the connection setup time.

Meanwhile, dynamic routing solution using AI is getting more attention. The more popular solutions are employing fuzzy logic and neural network, for the reason they can provide adaptive, model-free, real-time control to the user. Fuzzy logic put more focus on network controlling (CAC and UPC) and link cost calculation while neural network is mainly applied for optimal routing path finding. Integration of neuro-fuzzy in [10] provides the answer for the complex routing problem in ATM. Fuzzy logic with the capability of simultaneously maximizing several selected objectives produce the fuzzy link cost for each link as in [5]. The spiking neuron network will then compute the "optimal" route with minimum hop length using the associated fuzzy link cost.

As a result of the literature study, the proposed solution in this thesis will utilize fuzzy logic and spiking neuron network to implement the concept of both load balancing and bandwidth packing in ATM network to maintain guaranteed QoS while maximizing its cost-

effectiveness. The detail implementation of fuzzy link cost and spiking neuron network for the proposed solution are discussed in chapter 3.

Chapter 3

Methodology

The proposed solution implementation becomes concrete with the ideas and concepts inspired or learned from the research papers in previous chapter. The adopted concepts and ideas are load balancing, bandwidth packing, selected network state parameters (buffer and link utilization), integration of fuzzy link cost with spiking neuron network and usage of Designated Transit List (DTL).

The principle function of the proposed solution is to create a routing plan for the entire ATM network or ATM sub-network. The routing plan will try to balance the network load while trying to pack smaller connections at used links with the guaranteed quality to save bandwidth for future high requirement connections. The routing plan shall avoid using congested links that could cause performance degradation. The routing plan will influence every connection establishment by sending the source node a recommended DTL. The DTL is generated whenever there is a request for new connection. The DTL generation process relies on the network state metrics feedback from each node. The selected metrics are current buffer size on each link, average buffer changing rate on each link, available bandwidth on each link, load variation on each link to overall average network load and requested bandwidth size. Fuzzy link cost inference system and spiking neuron network are implemented to find the “optimal” route. Fuzzy inference system will process all gathered metrics to become fuzzy link cost. The fuzzy link cost will be passed to spiking neuron network to compute the shortest path route (in terms of minimum cost route). The spiking neuron network will provide all possible routes (route with the total cost within

acceptable range) and their total cost from the source node to destination node. This implementation is expected to reduce the burden of route computation from source node and reduces the crankback process possibility.

3.1 Fuzzy Link Cost

Fuzzy link cost is the implementation of fuzzy inference system in calculating link cost for each link based on the values from a set of selected metrics.

Routing in ATM is a complicated problem. Each link in ATM network is associated with multiple metrics, such as available bandwidth, average cell loss and buffer size. Some of these metrics have negative effects while some of them have positive effects on the connection performance. Furthermore, the resulting routing path depends on the requirements of the requested connection. One of the obvious example is the satisfaction of the requested bandwidth by a link would influence the result of routing path. The difficulties increase if it is multi-objectives routing. When cost effectiveness is one of the considerations, each connection should be assigned to links that just satisfy the requirements rather than over satisfying it.

Fuzzy logic is gaining recognition as a tool for handling imprecision problem with simplification and multi objectives optimization problem is one of the problems that cane be handled by fuzzy logic well. In conventional model, complicated mathematic formula with precise values is needed for calculating each link cost. The mathematic formula will become more complex when more metrics is taken into consideration. Applying fuzzy link cost could avoid using complicated mathematic formula. Fuzzy link cost can be easily modified or added with fuzzy rules to its existing rule-base when additional metric is put

into consideration. Fuzzy link cost could also model heuristic characteristics that are difficult to model using mathematical formula. The fuzzy feature in fuzzy link cost expands the feasible solution space by recognizing more qualified links as candidate links.

Fuzzy link cost is one of the key factors to identify the “optimal” route. The basic idea of fuzzy link cost is that its negative effect metrics will increase the link cost and positive effect metrics will decrease the link cost through defined fuzzy inference system. The minimum cost route will fulfill the objectives (with some trade off) indirectly.

3.2 Spiking Neuron Network

Spiking neuron network (SNN) has been considered as the third generation of neural network model in terms of neuron computation power [50]. According to [50], the first generation is McCulloh-Pitts neuron, known as perceptron. Original Hopfield neural network is the example of first generation. The second generation neuron is the neuron applying an activation function with a continuous set of possible output values for a weighted sum of input. Sigmoid function is one of the common activation functions in second generation neural network. Feedforward neural network is the example of second generation neural network model. Second generation neural network model has a learning feature on gradient descent such as backward propagation.

SNN have recently been shown to be computationally more powerful than first and second generation neural networks with respect to time and network complexity. SNN uses “time” as a resource for computation and communication. The timing of individual computation plays a key-role for computations in SNN. The firing time of a neuron encodes a value in the sense that an early firing of neuron represents a large value.

SNN is an efficient tool to calculate the shortest routing path from source node to destination node. Besides destination node, SNN is able to identify the shortest routing path from source node to other network nodes at the same time. This may helps to avoid re-computation when similar requirement request with different destination node. SNN has an additional feature to control tolerable cost/time to identify all candidate routes that fulfills the conditions. The capability to control tolerable cost/time is not found in any existing algorithm except for spiking neuron network. During the shortest routing path computation, SNN records details of all candidate routes such as cost/time difference for each link to allow optimal route selection when having candidate routes with same cost/time.

In the simplest (deterministic) model of spiking neurons, assume that a neuron v fires at time t wherever its potential $P_v(t)$ (modeling the electric membrane potential at the "trigger zone" of neuron v) reaches a certain threshold $\theta_v(t-t')$, where t' is the time of the most recent firing of v. This potential P_v is the sum of excitatory postsynaptic potentials (EPSP) and inhibitory postsynaptic potentials (IPSP). Figure 10 shows the shape of EPSP and IPSP. The postsynaptic potentials are the result of firing from other neuron u that is connected through a "synapse" to neuron v. The firing of a "presynaptic" neuron u at time s contributes to the potential P_v at time t on an amount that is modeled by the term $W_{u,v} \cdot R_{u,v}(t-s)$, which the weight $W_{u,v} \geq 0$ and a response function $R_{u,v}(t-s)$ between neuron u and neuron v.

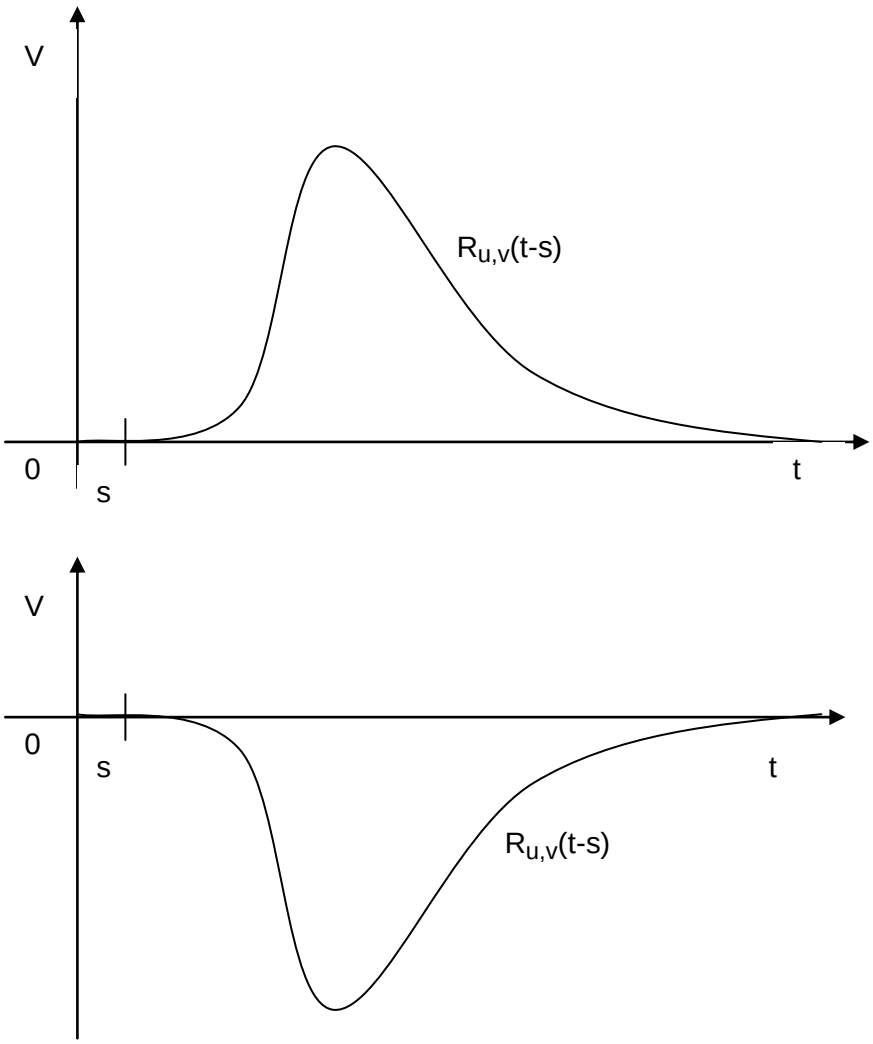

Figure 12: The shape of EPSP (above) and IPSP (below)

In stochastic or noisy version of the SNN model, the difference $P_v(t) - \theta_v(t - t')$ governs the probability that neuron v fires at time t. The choice of exact firing times is left up to some unknown stochastic processes. The noisy version SNN is basically identical with spike response model in [45] and [46].

The weight $W_{u,v} \geq 0$ reflects the strength of the synapse (called efficacy in neurobiology) between neuron u and neuron v. $W_{u,v}$ could be replaced by a function $W_{u,v}(t)$ for adding

in learning ability. The restriction of $W_{u,v}$ to non-negative values is motivated by the assumption that a biological synapse is either " excitatory" or "inhibitory" only, and it does not changes its sign in the course of a "learning" process.

If a neuron v has fired at time t', it will not fire again for a few m_{sec} after t' no matter how large its current potential P_v (t) is (this is known as absolute refractory period). Then for a few further m_{sec} it is still "reluctant" to fire, a firing requires a large value of P_v(t) than usual (this is known as relative refractory period). Both of these refractory effects are modeled by a suitable threshold function $\theta_v(t-t')$.

A formal spiking neuron network introduced in [51] and [49] consist of a finite set V of spiking neurons, a set $E \in v \times v$ of synapses, a weight $W_{u,v} \geq 0$ and a response function $R_{u,v}$ for each synapse (u, v) $\in$ E, and a threshold function θ_v for each neuron v$\in$V. If F_u is the set of firing times of a neuron u, then the potential at the "trigger zone" of neuron v at time t is given by

$$P_v(t) = \sum_{u:(u,v)\in E} \sum_{s\in F_u : s<t} W_{u,v} \cdot R_{u,v}(t-s)$$

For some specified subset of input neurons $V_{input} \subseteq V$ one assumes that the firing time, F_u, for neurons $u \subseteq V_{inputs}$ are not defined by the preceding convention, but are given from the outside. The firing time, F_v, for all other neurons v $\in$ V are determined by the previously described rules, and the output of the network is given in the form of spike train for the neurons v in a specified set of output neurons $V_{output} \subseteq V$.

3.3 The proposed solution model

The proposed fuzzy-neural integration routing model is as describe in figure 11. The proposed model uses metrics' value feedback from every network node and also requirements of requested connection as input for the routing computation. These metrics are current buffer size, buffer increase rate, bandwidth consumption, and requested bandwidth. Load variance and link compatibility are derived from bandwidth consumption, requested bandwidth and maximum bandwidth. Load variance is a measurement of the differences between each link's bandwidth consumption and the mean of bandwidth consumption. The load variance will have the positive or negative sign to indicate it is more or less than the mean. Link compatibility is a measurement on the ability of a link to support the requested bandwidth. The link compatibility will have the positive or negative value to indicate the link is capable to accept the request or not.

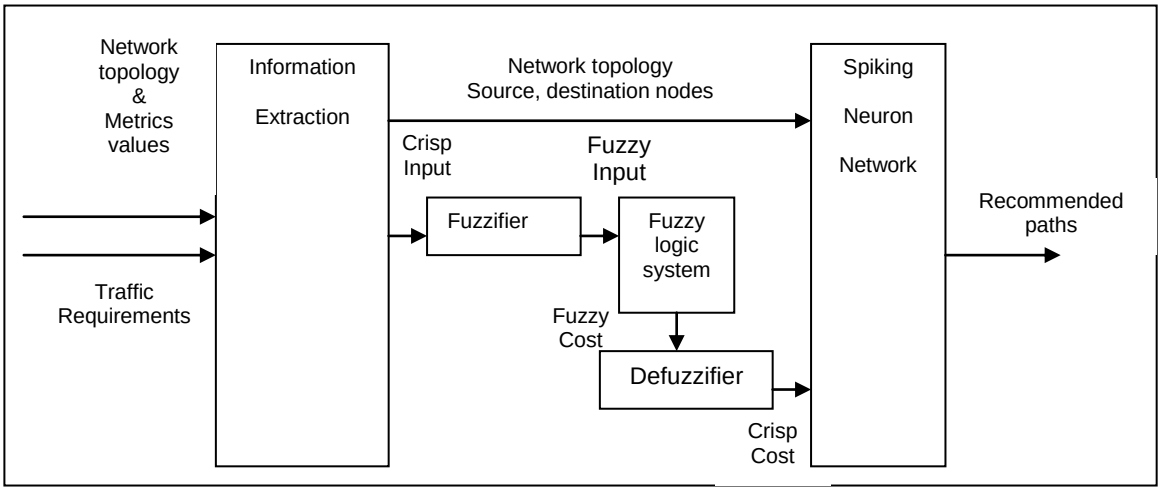

Figure 13: Proposed Solution Model

3.3.1 Fuzzy Link Cost

The proposed model consists of two parts. The first part is fuzzy link cost inference system. Inputs of the fuzzy inference system are metrics stated above. Every metric is fuzzified into linguistic variable for the use of fuzzy inference system. Each linguistic variable is associated with a membership function. In the proposed solution model, trapezium membership function is chosen for all linguistic variables. Trapezium membership function has the advantage that it can be easily transformed into triangular membership function or maintains as a trapezium membership function. Trapezium membership function has 4 parameter points. Transformation of trapezium membership function into triangular membership function could be done by setting the second and third parameter points to the same value.

The following are the membership function configuration and calculation for all the metrics:

Buffer queue metric has three linguistic variables, which are short, medium and long. The membership functions are:

short : trapezium(-45 –5 10 40)

medium: trapezium(10 45 60 90)

long : trapezium(55 85 100 150)

The formula to calculate the buffer queue (calculate the buffer utilization in percentage) is

$$\frac{Buffer_{utilized} \times 100}{Buffer_{Max}}$$

where $Buffer_{utilized}$ is buffer utilized at the link and

$Buffer_{Max}$ is the maximum buffer size.

Queue delta has three linguistic variables which are negative, small positive and large positive. The membership functions are:

negative : trapezium(-50 –10 10 20)

small positive: trapezium(0 20 70 90)

large positive : trapezium(50 80 100 150)

Queue delta calculates the difference of buffer queue size based on moving average (calculate the difference of buffer queue size in percentage), and starts calculate at second reading time. The calculation of queue delta is

$$\frac{\left(Queue_{average,t} - Queue_{average,t-1}\right)\times 100}{Buffer_{Max}}$$

where Queue average, t is buffer queue average size at time t,

Queue average, t-1 is buffer queue average size at time t-1, and

Buffer Max is the maximum buffer size.

The calculation of buffer queue average is

$$Queue_{average,t} = Queue_{average,t-1} \times Weight_{average} + Buffer_{utilized,t} \times Weight_{utilized}$$

where Queue $_{\text{average, t-1}}$ is the buffer queue average size at time t-1,

Buffer $_{\text{utilized, t}}$ is the current buffer size,

Weight $_{\text{average}}$ is the weight for average buffer size, and

Weight $_{\text{utilized}}$ is the weight for current buffer size.

Calculating the changes of buffer size in moving the average is for the purpose of reducing the solution model's sensitivity to traffic fluctuations.

Requested bandwidth has three linguistic variables. They are small, medium and large. The membership functions are:

small : trapezium(-45 0 0 50)

medium : trapezium(15 45 55 85)

large : trapezium(50 100 100 150)

The requested bandwidth is normalized (calculate the requested bandwidth in percentage) by

$$\frac{Bandwidth_{\mathrm{Re}\ qeusted} \times 100}{Bandwidth_{Max}}$$

where Bandwidth $_{\text{Requested}}$ is the requested bandwidth, and

Bandwidth $_{\text{Max}}$ is the maximum bandwidth.

Link compatibility is associated with three linguistic variables. The linguistic variables are negative, zero and positive. The membership functions are:

negative : trapezium(-190 –110 –45 15)

zero : trapezium(-15 0 0 15)

positive : trapezium(-15 45 100 190)

Link compatibility measures the ability of a link to support the requested bandwidth. If the calculated result is positive, this indicates that the link is capable in accepting the request. The calculation (calculate link compatibility in percentage) is

$$\frac{\left(Bandwidth_{Max} - Bandwidth_{Utilized} - Bandwidth_{\mathrm{Re}\ quested}\right) \times 100}{Bandwidth_{Max}}$$

where Bandwidth $_{\text{Max}}$ is the maximum bandwidth,

Bandwidth $_{\text{Utilized}}$ is the utilized bandwidth on the link, and

Bandwidth $_{Requested}$ is the requested bandwidth.

Load variance has four linguistic variables: large negative, negative, positive and large positive. The membership functions are:

large negative : trapezium(-145 –100 –50 -10)

negative : trapezium(-15 –5 –5 5)

positive : trapezium(-5 5 5 15)

large positive : trapezium(10 50 100 145)

The load variance measures the difference between the utilized bandwidth on the link and the mean of utilized bandwidth. The calculation (calculate in percentage) is

$$\frac{(Bandwith_{utilized} - Bandwidth_{Mean}) \times 100}{Bandwidth_{Max}}$$

where Bandwidth $_{utilized}$ is the utilized bandwidth on the link,

Bandwidth $_{Mean}$ is the mean of utilized bandwidth, and

Bandwidth $_{Max}$ is the maximum of bandwidth.

The output of the fuzzy inference system is the fuzzy link cost. The Cost has three linguistic variables: low, medium and high. The membership functions are:

low : trapezium(0 0 5 35)

medium : trapezium(15 35 65 85)

high : trapezium(65 95 100 100)

Table 1 contains the rule base used in the fuzzy link cost inference system. The rules are logically separated into three sections. The first section measures the degree of congestion reflected by the buffer size and the changes of buffer size. The second section determines the ability of the link to support the requested bandwidth. The third section tries

to balance the load while packing them “compactly”. Bandwidth packing is useful in avoiding bandwidth fragmentation. Packing small bandwidth connection to higher utilized link could prepare lower utilized link for larger bandwidth connection. The fuzzy control method that is implemented is the most widely used approach, which is the Mamdani approach. The selected defuzzification method to produce crisp value of COST is Center of Area.

Table 1: Fuzzy Link Cost Inference System Rule base

NO	BUFFER QUEUE	QUEUE DELTA	COST	WEIGHT
1	not long	negative	low	0.5
2	long	negative	medium	0.5
3	-	large positive	high	0.5
4	short	small positive	low	0.5
5	medium	small positive	medium	0.5
6	long	small positive	high	0.5
	LINK COMPATIBILITY		**COST**	**WEIGHT**
7	negative	-	high	1
8	zero	-	medium	1
9	positive	-	low	1
	LOAD VARIANCE	**REQUESTED BANDWIDTH**	**COST**	**WEIGHT**
10	large negative	-	low	0.5
11	large positive	-	high	0.5
12	negative	small	high	0.5
13	negative	medium	medium	0.5
14	negative	large	low	0.5
15	positive	small	low	0.5
16	positive	medium	medium	0.5
17	positive	large	high	0.5

In the first section of rule base, the Queue Delta metrics has bigger influence than the Buffer Queue Size. Whenever Queue Delta has the value of large positive, the Cost is high and whenever Queue Delta has the value of negative, the Cost is medium or low. The Buffer Queue has its determining influence when the Queue Delta has stable changes.

In the second section of rule base, the Link Compatibility has reverse effect on the Cost with a higher weight. When the Link Compatibility is negative, the Cost is high and when the Link Compatibility is positive, the Cost is low. Link Compatibility has higher weight compared to other section because it is the primary determination factor on whether to accept the request on the link or not.

In the third section of rule base, the Load Variance has higher influence than the Requested Bandwidth. When the Load Variance is large negative or large positive, the Cost is low or high respectively. When the Load Variance is negative, the Requested Bandwidth has reverse relation with the Cost. When the Load Variance is positive, the Requested Bandwidth has direct relation with the Cost.

3.3.2 Spiking Neuron Network

The second part of the proposed model is the spiking neuron network. The spiking neuron network is implemented to solve the shortest path problem. The neurons in the spiking neuron network are arranged according to the ATM network topology. Each neuron in the neural network represents one network node and each synapse represents a link between the ATM network nodes. A link cost value, which is the output from the fuzzy link cost inference system, is associated with a synapse. The spiking neuron network utilizes time as a computational resource which allows the translation of the link cost into the propagation delay time (in millisecond), t_{ij}, between the firing neuron i and neuron j. In the proposed spiking neuron network model, only excitatory postsynaptic potential is considered.

The proposed spiking neuron network is a simple, modified and deterministic spiking neuron network. The proposed spiking neuron network assumes that single postsynaptic potential is sufficient to trigger the postsynaptic neuron and activate the synapse. Whenever a postsynaptic potential arrives at a postsynaptic neuron through a synapse and if the synapse has never been activated in current learning cycle, the synapse will be activated. If the synapse is activated within the correlation time, t_{cor}, the synapse's weight, w, increases, or else it decreases. The t_{cor} time is calculated from the moment of the postsynaptic neuron firing in current learning cycle. The first postsynaptic potential is the one that determines the firing of the neuron. The firing of excitatory neurons activates the paired of inhibitory neuron (next excitatory neuron), which in turn inhibits the excitatory neuron for a period of time that is longer than the longest path. In such a case, the inhibitory period will become the time needed to travel through the longest path. Hence, each neuron node fires only once in a learning cycle. The t_{cor} time for synapse activated by the first postsynaptic potential is 0 because the firing time is same as the activation time of synapse. The correlation time, t_{cor}, starts with a large value, which slowly decreases throughout the learning process. The minimum value for correlation time is the minimum tolerable cost for all candidate routes for shortest path.

Initially, all synapses are assigned the same weight, which is 1 in this proposed model. While going through learning cycles, the weight of each synapse is updated by a learning function. (Please note that, during the learning cycles two values are adjusted; weight may increase or decrease depending on correlation time and correlation time is decreased for every cycle.)

The learning function that adjusts the weight of synapse uses temporal correlation function in [37]. The learning function is as follows:

$$w_{ij}(t+1) = w_{ij}(t) \times (1 + \alpha \times s(t))$$

where α is the learning rate (α<1),

t is the delay time, and

s(t) is the temporal correlation function that describes the correlation between neuron i and j.

The temporal correlation function is

$$s(t) = (1+y)\exp\left(-k\left(\frac{t}{t_{cor}}\right)^2\right) - y$$

where t is the current time,

t_{cor} is the correlation time,

k is $k = \ln\left(1+\frac{1}{y}\right)$, and

y is the maximum negative excursion (1).

The temporal correlation learning function allows variable increment or variable decrement on synapse weight according to the difference of postsynaptic potential arriving time and correlation time. Figure 12 shows the shape of temporal correlation function. As displayed in Figure 12, when the delay time is less than the correlation time the synapse's weight is rewarded else the synapse's weight will decrease.

A learning cycle starts by initiating a spike at the "input" neuron and ends when all the neurons in the network have fired and all postsynaptic potentials are terminated. In the proposed model, the input neuron is the source node of the requested connection. After a number of cycles, depending on the value of the learning rate, some synapses' weight become larger, until it reaches a saturation value, while other synapses' weight decrease and disappear at the end. As a result, the survivor synapse connecting each neuron

becomes the candidate route with the minimum or acceptable link cost. In the proposed solution model it is the “optimal” path. The produced candidates route are then send to network nodes in the form of DTL.

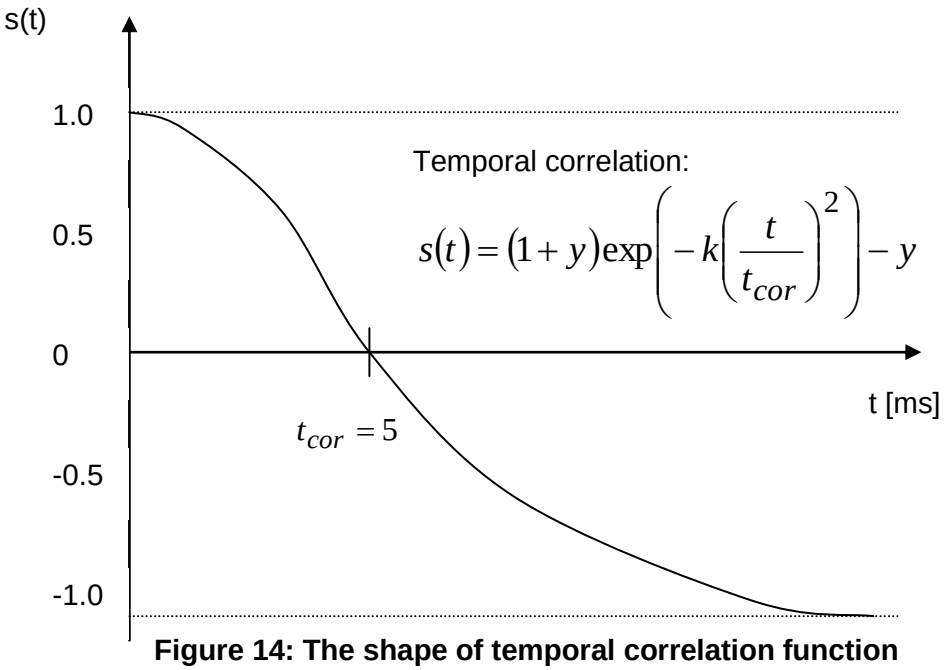

Figure 14: The shape of temporal correlation function

3.4 Conclusion

The principle function of the proposed solution is to create a routing plan for the entire ATM network or ATM sub-network. The routing plan will try to balance the network load while trying to pack smaller connections at used links with the guaranteed quality for future high requirement connections. Fuzzy link cost inference system and spiking neuron network are implemented to find the “optimal” route. The fuzzy link cost inference system will process all gathered metrics to calculate the fuzzy link cost. The fuzzy link cost will then be passed to spiking neuron network to compute the shortest path route.

Each link in an ATM network is associated with multiple metrics. Some of them have negative effects while the others have positive effects. Furthermore, the requirements of requested connection determine the participation of a link in route selection. The difficulties increase when it is multi objectives routing. Fuzzy logic is a recognized tool for handling multi-objectives optimization problem. Fuzzy link cost inference system is easy to model to include heuristic rules. The fuzzy feature in fuzzy link cost also increases the solution space by recognizing more qualified links as candidate link.

Spiking neurons network (SNN) has been considered as the third generation of neural network model in terms of the neuron computational power. SNN uses “time” as a resource for computation and communication. The timing of individual computation plays a key-role for computations in SNN. The firing time of a neuron encodes a value in the sense that an early firing of neuron represents a large value. SNN is an efficient tool to solve the shortest routing path from source node to destination node. SNN has the capability to control tolerable cost/time that is not found in existing algorithm to increase candidate routes. During the shortest routing path computation, SNN records details of all candidate routes to allow optimal route selection when there is more than one candidate route.

In the proposed model, the selected reference metrics are current buffer size, buffer increase rate, bandwidth consumption and requested bandwidth. From these metrics, the fuzzy link cost derives its link cost based on congestion level, acceptance of requested connection and load distribution. The fuzzy link cost uses buffer size and buffer increase rate as indicators for measuring congestion level, using available bandwidth and requested bandwidth to determine the request acceptance level and using load variance and requested bandwidth to plan on the load distribution. The fuzzy link cost is then passed to spiking neuron network to compute the shortest path problem. The neurons in the spiking neuron network are arranged according to the ATM network topology. Each neuron in the

neural network represents one network node and each synapse represents a link between the ATM network nodes. A link cost is associated with a synapse. The link cost is considered as the propagation delay time (in millisecond) between the firing neuron and its paired neurons. The spiking neuron network finds the shortest path by going through learning cycles. While going through learning cycles, the weight of each synapse is updated by a learning function and the correlation time decreases for every cycle. The correlation time is a parameter in the temporal correlation learning function that enables variable increases or decreases on synapse weight according to the firing time of presynaptic neuron. After a number of cycles, some synapses' weight become larger while other synapses' weight decrease and disappear at the end. As a result, the survivor synapse connecting each neuron is the candidate route with the minimum link cost.

Intelligent routing implementation by integrating fuzzy link cost and spiking neuron network is able to optimize multiple objectives in parallel and solving the shortest path problem with maximum solution space. The produced candidate routes are then send to network nodes in the form of DTL. An analysis of the simulation results is presented in the following chapter.

Chapter 4

Discussion of Results

Continuing from the previous chapter, the next step is to conduct an experiment to find out the performance of the proposed theory and model. A control model is designed to work as a benchmark to compare against the proposed theory and model. The chosen control model is an integration of crisp link cost calculation with Djikstra algorithm. The following document will refer crisp-Djikstra model as the control model and fuzzy-neuro model as the proposed model. Both control model and proposed model are presented to the experiment environment to gather performance data. Both models will select the "optimal" route for a few connection requests. The selected performance metrics for comparison are the number of accepted connections and cell loss ratio.

This chapter will first introduce the crisp-Djikstra model in detail. This follows with the explanation of the experiment's objectives and experiment settings. The experiment settings include experiment topology and its environment values. After the explanation of experiment details, followed by an experimental result discussion. Continuing from experimental result discussion will be on model comparison for further explanation on the underlying concepts. Finally, this chapter will end with the comparison.

4.1 Control Model

The control model is an integration of two main components: crisp link cost and Djikstra shortest path algorithm. Crisp link cost is a mathematical link cost calculation for network

links. Djikstra shortest path algorithm is then applied to find the shortest path from the source node to the destination node using the calculated link cost. The crisp link cost is designed to calculate the link cost with similar considerations of the proposed model (congestion level, link compatibility and load distribution) to make it a fair comparison.

Crisp link cost is the calculation of link cost between two adjacent nodes using conventional mathematic formula, $f_{cost}(c,v,b)$. The formula of $f_{cost}(c,v,b)$ is

$$f_{\cos t}(c,v,b) = \begin{cases} 0 & if \;\; f_{aoc}(c) < 0 \\ \dfrac{f_{buv}(v)w_1 + f_{bu}(b)w_2}{w_1 + w_2} & if \;\; f_{aoc}(c) \geq 0 \end{cases}$$

where W_1, W_2 are weights for $f_{buv}(v)$ and $f_{bu}(b)$ respectively,

$f_{aoc}(c)$ is the function calculating the acceptance of connection,

$f_{bu}(b)$ is the function calculating the buffer utilization, and

$f_{buv}(v)$ is function to calculate link cost of bandwidth utilization variance.

The $f_{buv}(v)$ is a sigmoid function with the formula,

$$\frac{f}{e^{-kv/d}}$$

where f is center of the sigmoid function,

k and d are variables that determine the steepness of the sigmoid curve and

v is the normalized variance of each link utilization.

The formula for link utilization variance normalization, v (calculated in percentage) is

$$\frac{(Bandwidth_{Utilized} - Bandwidth_{Mean}) \times 100}{Bandwidth_{Max}}$$

where Bandwidth $_{Utilized}$ is utilized bandwidth,

Bandwidth $_{Mean}$ is mean of utilized bandwidth for all possible routes to a node, and

Bandwidth $_{Max}$ is the maximum bandwidth.

The formula for $f_{aoc}(c)$ (calculated in percentage) is

$$\frac{\left(Bandwidth_{Max} - Bandwidth_{Utilized} - Bandwidth_{\mathrm{Re}\,quested}\right)\times 100}{Bandwidth_{Max}}$$

where Bandwidth $_{Max}$ is the maximum bandwidth,

Bandwidth $_{Utilized}$ is the utilized bandwidth, and

Bandwidth $_{Requested}$ is the requested bandwidth by the connection.

The formula for $f_{bu}(b)$ is

$$\frac{Buffer_{Utilized}\times 100}{Buffer_{Max}}$$

where Buffer $_{Utilized}$ is buffer utilized at the network node and

Buffer $_{Max}$ is the maximum buffer size.

Djikstra algorithm is a well-known shortest path finding algorithm from a source node to all other nodes. The Djikstra algorithm uses the network topology and the calculated link cost to find the shortest path. In the case of control model, the cost of each link is considered as the distance between two nodes.

4.2 Experiment objective

The experiment objective is to identify the performance of the proposed fuzzy-neuro model in ATM network. A control model is prepared to work as a benchmark to measure the

performance of the proposed fuzzy-neuro model. The performance measurement depends on the following two metrics:

i. Number of accepted connection. In telecommunication industry, each successful accepted connection is considered as revenue to network operator and each rejection of connection request is equivalent to a lost. All network operators want to maximize the number of accepted connection to generate more revenue with their existing network equipments. Therefore, the number of accepted connection is suitable to be one of the network performance indicators.

ii. Cell loss ratio. Cell loss ratio is another suitable metric to measure network performance. It measures the possibility of a cell being successfully transmitted over the network. Cell loss ratio calculates the number of cell lost over total transmitted cells. The main reason of lost cells is due to the buffers are full or occurrence of network congestion. The cell loss ratio is higher when the connection is passing through a congested area.

The calculated link cost from both crisp-Djikstra and fuzzy-neuro models could never do cross model comparison. The calculated link costs from both models are relative costs that are only effective within each model. The calculated link cost is only meant to identify the relative “optimal” route.

4.2.1 Experiment Environment

4.2.1.1 Topology

The experiment network topology is designed as in figure 15. Each node in the network is labeled with a number. In this experiment network topology, each link between two nodes is a single direction connection from the lower number node to the higher number node. Each link between two nodes has 155 Mbps bandwidth. The experiment network topology uses three types of network nodes: source node, routing node and destination node. The rectangles in figure 13 are routing nodes that are responsible in transferring data from source node to destination node. The round rectangles are destination nodes that receive and terminate data transfer. Cell loss ratio is calculated at the destination nodes. The circles are source nodes that transfer data with variable data rate to destination nodes. The number stated in each round rectangle is continued according to the number in rectangle. Two routing node with label x are the routing nodes that are not within the scope of experiment. These two routing nodes are added for the purpose to create the network traffic load accumulated from source node 1, 2, 11 and 12. The routing nodes x are virtually transparent to the experiment. The routing nodes x are not counted in the hop count. The bottleneck of this experiment network topology is at the links connecting node 7 (from node 5 to node 7 and from node 6 to node 7). These two links are hereafter known as the bottleneck links. The bottleneck links are intentionally created to demonstrate the intelligence of the routing algorithm.

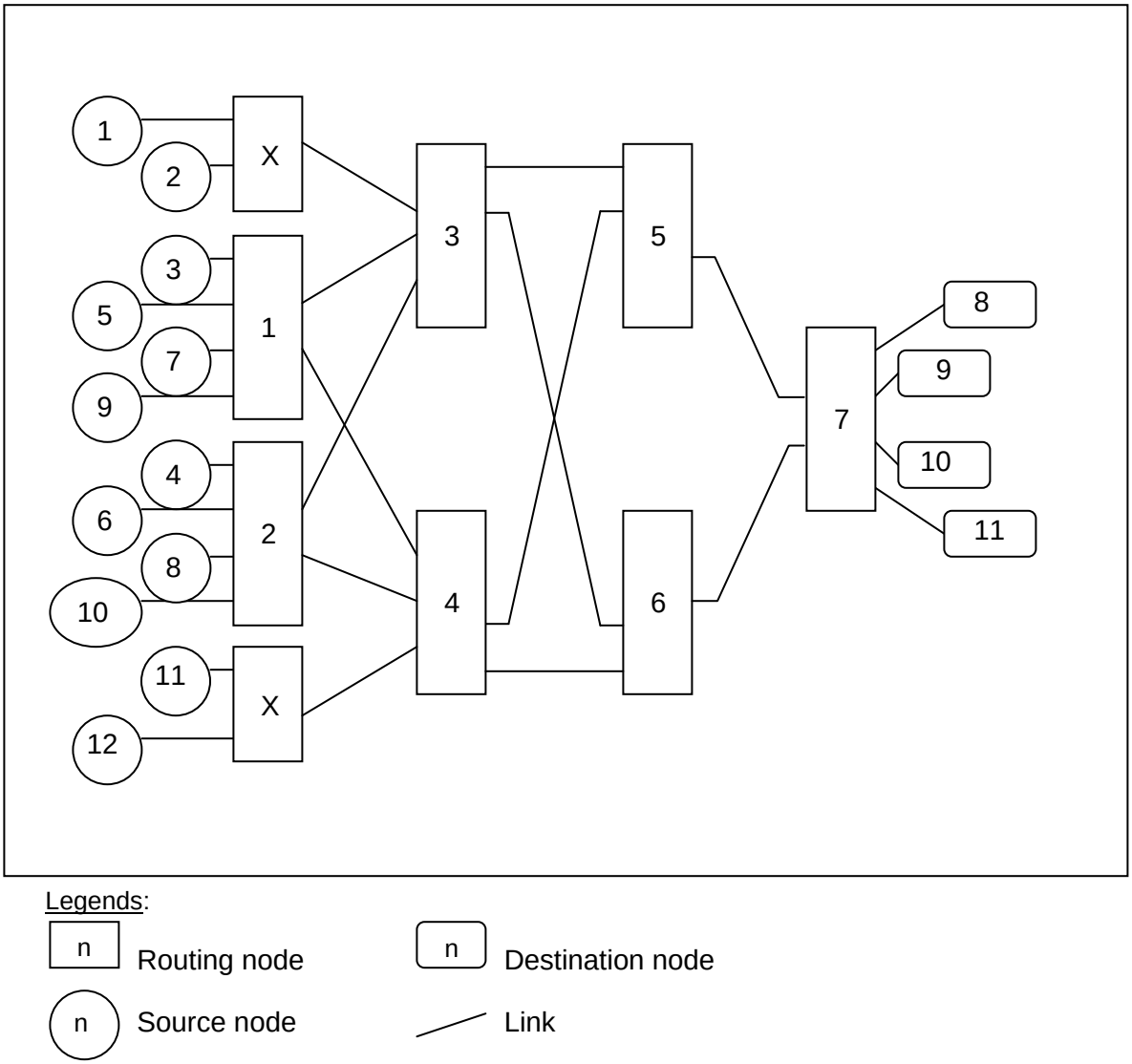

Figure 15: Experiment Topology

4.2.1.2 Experiment Settings

The experiment is designed to have 12 source nodes transferring data to 4 destination nodes, passing through the routing node 7 as in Figure 13. The source nodes are the circles labeled from 1 to 12. The destination nodes are the round rectangles labeled from 8

to 11. Among 12 source nodes, there are 4 categories of source node differentiated by their data transfer rate: 35Mbps, 30Mbps, 20Mbps and 10Mbps.

Table 2: Connection Settings

Source Node	Destination Node	Transfer Data Rate (Mbps)	Start Time (s)	End Time (s)
1	8	35	0.00001	6.00000
2	8	35	0.00003	6.00000
3	11	30	0.00006	6.00000
4	9	20	0.00009	6.00000
5	9	20	0.50000	6.00000
6	9	20	1.00000	6.00000
7	10	10	1.50000	6.00000
8	10	10	2.00000	6.00000
9	10	10	2.50000	6.00000
10	11	30	3.00000	6.00000
11	8	35	3.50000	6.00000
12	8	35	4.00000	6.00000

As stated in the above table, 4 of the 12 connections from source 1, 2, 11 and 12 (transferring 35Mbps each) to destination 8 are created to load the network with almost 50% of network bandwidth. These 4 connections are traveling through its defined route to ensure the network load is equally distributed to entire network. Connections from source 3 to source 10 are connections that will determine their route through the routing algorithm in the control model or the proposed model. Destination 11 will collect all cells with data rate 30Mbps (from source 3 and source 10), destination 9 will collect all cells with data rate 20Mbps (from source 4, 5 and 6) and destination 10 will collect all cells with data rate 10Mbps (from source 7, 8 and 9).

Each routing node has 4 buffers, 1 buffer for each input link. The buffer size of each buffer is 1000 units of ATM cells, which is 53 Kbytes each. Whenever the input buffer becomes full, the routing node will discard the next incoming ATM cells until the input buffer is available to accept cells.

In this experiment, the Connection Admission Control and Usage Policy Control are not implemented. The assumption is that all accepted connections have the guaranteed resources and adhere to the agreed data rate for the lifetime of the connection.

This experiment is repeated twice (1 for control model and 1 for proposed model) to gather the performance of the control model and the proposed model with the network load of 70Mbps at the bottleneck links (from node 5 to node 7 and from node 6 to node 7). The focus of the experiment is on the routing path calculation for connections from source 3 to source 10 and cell loss ratio at all destination nodes. Although destination 8 is collecting cells transmitted from source 1, 2, 11 and 12 (routing path not determined through routing calculation), its cell loss ratio is also taking part in the result analysis. During the experiment result analysis stage, I will discuss the accepted connection with its routing path and the cell loss ratio for each traffic types. Continue from the discussion is a cross model experiment result comparison to determine the pros and cons of each model.

4.3 Experimental results

Every connection from source node to destination node is given a connection id for the purpose to identify the origin and the destination. The connection id is unique throughout the entire experiment. In the following experiment result analysis, the connection id is used to identify each connection instead of the source node id.

Table 3: Connection ID of Source Node

Source Node	Connection ID
1	10
2	20
3	30

4	40
5	50
6	60
7	70
8	80
9	90
10	100
11	110
12	120

Since the data traffic from source 1, 2, 11 and 12 are to create network load in the network, their routing path is pre-determined before the experiment begins.

Table 4: Pre-Determined Routing Path of Connection 10, 20, 110 and 120

Connection ID	Routing Path	Transfer Data Rate (Mbps)
10	3, 5, 7, 8	35
20	3, 6, 7, 8	35
110	4, 5, 7, 8	35
120	4, 6, 7, 8	35

Before source 3 (the first source node requesting for connection through routing algorithm), the network is loaded with 70Mbps at each bottleneck link (link 5-7 and 6-7) and 35Mbps at the other links (link 3-5, 3-6, 4-5 and 4-6). The routing algorithm needs to intelligently distribute 150 Mbps of traffic (total from connection 30 to connection 100).

4.3.1 Results of Control Model

For the control model, before a request arrives from connection 30, the bandwidth utilization of link 5-7 and link 6-7 are 70Mbps each. The following table is the assigned routing path for connections from connection 30 to connection 100.

Table 5: Assigned Routing Path for Control Model's Connections

Connection ID	Routing Path	Transfer Data Rate (Mbps)
30	1, 3, 5, 7, 11	30
40	2, 3, 6, 7, 9	20
50	1, 4, 6, 7, 9	20
60	2, 4, 5, 7, 9	20
70	1, 4, 6, 7, 10	10
80	2, 4, 5, 7, 10	10
90	1, 3, 6, 7, 10	10
100	-	30

The control model computes the routing path based on load balancing concept. After accepting connection 90, the routing algorithm rejects connection 100 because it is unable to allocate sufficient resources for the request.

Table 6: Bandwidth Utilization Changes for Control Model

Timeline	Request Bandwidth (Mbps)	Bandwidth Utilization at every link (Mbps)					
		3-5	3-6	4-5	4-6	5-7	6-7
Before 30	0	35	35	35	35	70	70
30[1]	30	65	35	35	35	100	70
40[2]	20	65	55	35	35	100	90
50[3]	20	65	55	35	55	100	110
60[4]	20	65	55	55	55	120	110
70[5]	10	65	55	55	65	120	120
80[6]	10	65	55	65	65	130	120
90[7]	10	65	65	65	65	130	130
100[8]	30	65	65	65	65	130	130
After 100	0	65	65	65	65	130	130

When connection 30 requests for connection, the network load is evenly distributed. The routing algorithm randomly selects a route for connection 30. The routing algorithm then assigns route (2, 3, 6, 7, 9) to connection 40 to balance the network load at bottleneck links (link 5-7 and 6-7). When accepting request from connection 50, 60 and 70, the routing algorithm distributes the network load among the 3-5, 3-6, 4-5 and 4-6 links evenly. After connection 70, the bottleneck links (link 5-7 and 6-7) are evenly loaded with 120 Mbps. During the acceptance of connection 80 and 90, the routing algorithm distributes the network load of 130 Mbps evenly to the bottleneck links (link 5-7 and 6-7) and 65 Mbps to the links at 3-5, 3-6, 4-5 and 4-6. As a result of even distribution of network load to the

[1] Connection 30 is accepted at link 3-5 and 5-7.

[2] Connection 40 is accepted at link 3-6 and 6-7.

[3] Connection 50 is accepted at link 4-6 and 6-7.

[4] Connection 60 is accepted at link 4-5 and 5-7.

[5] Connection 70 is accepted at link 4-6 and 6-7.

[6] Connection 80 is accepted at link 4-5 and 5-7.

[7] Connection 90 is accepted at link 3-6 and 6-7.

[8] Connection 100 is rejected.

bottleneck links, the control model is unable to accept connection 100 that is requesting a 30 Mbps bandwidth.

Table 7: Average Cell Loss Ratio of Connections for Control Model

Destination Node	Expectation	Actual Received	Differential	Average CLR
8	225258	176430	48828	0.2168
9	63612	52995	10617	0.1669
10	21434	16909	4525	0.2111
11	52772	38512	14160	0.2683

Every destination node in the experiment gathers transmitted data with a specified data rate. Destination 8, 9, 10 and 11 collect cells transmitting from data source with 35 Mbps, 20 Mbps, 10 Mbps and 30 Mbps respectively. Therefore, the number of received cell at each destination can be used to calculate the average cell loss ratio (CLR) for a specific data rate.

From the above table, 20 Mbps data traffic has better performance than other data traffic. It has lower average CLR compare to other data traffic. In overall, the control model has 0.2158 average CLR with standard deviation 0.0415 for the entire experiment.

4.3.2 Results of Proposed Model

Similar to the control model, before a request from connection 30, the bandwidth utilization for link 5-7 and link 6-7 are 70 Mbps each. The assigned routing paths for connections from connection 30 to connection 100 are stated in the following table.

Table 8: Assigned Routing Path for Proposed Model's Connections

Connection ID	Routing Path	Transfer Data Rate (Mbps)
30	1, 3, 5, 7, 11	30
40	2, 3, 5, 7, 9	20
50	1, 3, 6, 7, 9	20
60	2, 3, 6, 7, 9	20
70	1, 3, 6, 7, 10	10
80	2, 4, 5, 7, 10	10
90	1, 3, 5, 7, 10	10
100	2, 4, 6, 7, 11	30

Unlike the control model, the proposed model adopts both load balancing and bandwidth packing methodology in its routing path computation. The routing algorithm for the proposed model is able to accept all connections together with their requested resources.

Table 9: Bandwidth Utilization Changes for Proposed Model

Timeline	Request Bandwidth (Mbps)	Bandwidth Utilization at every link (Mbps)					
		3-5	3-6	4-5	4-6	5-7	6-7
Before 30	0	35	35	35	35	70	70
30[9]	30	65	35	35	35	100	70
40[10]	20	85	35	35	35	120	70
50[11]	20	85	55	35	35	120	90
60[12]	20	85	75	35	35	120	110
70[13]	10	85	85	35	35	120	120
80[14]	10	85	85	45	35	130	120
90[15]	10	95	85	45	35	140	120
100[16]	30	95	85	45	45	140	150
After 100	0	95	85	45	45	140	150

Similar to control model, before connection 30, the network load is evenly distribute with 70 Mbps at link 5-7 and link 6-7. When connection 30 requests for connection, the proposed routing algorithm randomly selects a routing path for it. The proposed routing algorithm assigns routing path (2, 3, 5, 7, 9) to connection 40 under the influence of bandwidth packing which leaves more bandwidth at link 6-7. Continue from connection 40, connection 50 is assigned to go through link 6-7 based on the load balancing concept. The assignment of routing path for connection 40 and 50 has demonstrated both load balancing and bandwidth packing characteristics. The following connection 60 and connection 70 are assigned to go through link 6-7 to balance the network load at

[9] Connection 30 is accepted at link 3-5 and 5-7.

[10] Connection 40 is accepted at link 3-5 and 5-7.

[11] Connection 50 is accepted at link 3-6 and 6-7.

[12] Connection 60 is accepted at link 3-6 and 6-7.

[13] Connection 70 is accepted at link 3-6 and 6-7.

[14] Connection 80 is accepted at link 4-5 and 5-7.

[15] Connection 90 is accepted at link 3-5 and 5-7.

[16] Connection 100 is accepted at link 4-6 and 6-7.

bottleneck links (link 5-7 and 6-7). After connection 70, notice that links 3-5 and 3-6 are evenly loaded with 85 Mbps of traffic, links 4-5 and 4-6 are evenly loaded with 35 Mbps and links 5-7 and 6-7 are evenly loaded with 120 Mbps. Connection 80 is being randomly assigned to go through link 5-7 but is designed to go through link 4-5. The selection of link 4-5 for connection 80 is due to the policy of "balancing the network load when variance is big". Bandwidth packing then influences connection 90 to leave more resources at link 6-7. The proposed model is able to accept connection 100 with the routing path (2, 4, 6, 7, 11) due to the integration of load balancing with bandwidth packing concepts.

Table 10: Average Cell Loss Ratio of Connections for Proposed Model

Destination Node	Expectation	Actual Received	Differential	Average CLR
8	225258	163062	62196	0.2761
9	63612	52651	10961	0.1723
10	21434	13907	7527	0.3512
11	52774	52766	8	0.0002

As describe in control model, every destination in the experiment can be used for calculating the average cell loss ratio (CLR) for a specific data rate. In proposed model, the average CLR for each data rate has large differences. 35 Mbps data traffic has the best performance while 10 Mbps data traffic has the largest average CLR. In overall, the proposed model has 0.2 average CLR with standard deviation 0.1520 for the entire experiment.

4.3.3 Model comparison

For the control model, begin from connection 30 to connection 70, 100 Mbps network load is evenly distributed over the bottleneck links (link 5-7 and 6-7) under the influence of the load balancing concept. Meanwhile, network load at links 3-5, 3-6, 4-5 and 4-6 are evenly spread over them. The control model then continues to load balance the network load by assigning routing path (2, 4, 5, 7, 10) and (1, 3, 6, 7, 10) to connection 80 and 90 respectively, which is causing both links 5-7 and 6-7 to be loaded with 130 Mbps. Each of the bottleneck links (link 5-7 and 6-7) has 25 Mbps free, however, the requested bandwidth from connection 100 is 30 Mbps and that is 5 Mbps more than the available bandwidth. As a result, control model rejects the request from connection 10. The disadvantage of pure load balancing concept is that it causes a small fraction of bandwidth unused in every link. The difficulties in formulation of crisp mathematic link limit the flexibility of routing algorithm in adopting more than one concept. The limited routing algorithm, in some situation, fails to achieve the objective to maximize the network resources utilization.

For the proposed model, from connection 30 to connection 40, total request of 50 Mbps is packed onto the same partial routing path. (3, 5, 7). Continue from connection 40, the proposed model routing algorithm evenly distributes the network load to the partial routing path (3, 6, 7) for connection 50, 60 and 70. The switch in the applied concepts from bandwidth packing to load balancing happens when the fuzzy link cost detects a large variance between link 6-7 and the mean utilization. Before the request from connection 80, the network load at bottleneck links (link 5-7 and 6-7) are evenly loaded with 120 Mbps. Connection 80 and 90 are packed together at link 5-7 which saves 35 Mbps free

bandwidth at link 6-7. Connection 100 is accepted due to the intelligent action of the proposed model routing algorithm to save the 35 Mbps at link 6-7. As a result, the routing algorithm accepts all requests for connection by loading 140 Mbps at link 5-7 and 150 Mbps at link 6-7.

Table 11: Average Cell Loss Ratio Comparison of Control Model and Proposed Model

Destination Node	Data Traffic (Mbps)	Average CLR (Control Model)	Average CLR (Proposed Model)
8	35	0.2168	0.2761
9	20	0.1669	0.1723
10	10	0.2111	0.3512
11	30	0.2683	0.0002

Table 12: Cell Loss Ratio Statistical Values Comparison of Control Model and Proposed Model

	Control Model	Proposed Model
Overall Average CLR	0.2158	0.2000
CLR Standard Deviation	0.0415	0.1520

From table 11, proposed model has higher CLR compared to the control model except for the 30 Mbps data traffic. 10 Mbps data traffic has the largest difference among four data traffics. When comparing the average CLR of the proposed model to the control model according to table 11, one may conclude that the control model outperforms the proposed model. However, remember that the proposed model accepts an additional connection that is transmitting at 30 Mbps and the average CLR has no significant differences. Furthermore, from table 12, the proposed model has lower overall average CLR with higher standard deviation. This means that the proposed model has better overall performance with higher degradation risk than the control model.

The proposed model's artificial intelligence implementation is the main reason why it performs better than the control model. The proposed model integrates a few concepts

(load balancing, packing, congestion avoidance) into its implementation. The ability to integrate a few concepts that may contradict is what the control model lack of. Fuzzy logic enables the proposed model to integrate a few concepts easily by replacing complex numeric formulas with human understandable labels and rules. Besides fuzzy logic, spiking neuron network implemented in the proposed model is capable to find the shortest path from one source node to all destination nodes like Djikstra algorithm. However, spiking neuron network could find all alternative paths to all destination nodes. This provides the routing algorithm with more selection on choosing the “best” route. If under any circumstances that a selected route is rejected, an alternative path could be found without recalculation.

4.4 Conclusion

In chapter 3, fuzzy-neural model (the proposed model) is introduced and discussed to solve the dynamic routing problem. The proposed model is expected to perform better than any conventional dynamic routing methodology for its artificial intelligence implementation. Theories and concepts implemented in proposed model are individually proven more effective in some studies [5], [20], [37], [39]. However, those theories and concepts are to be proven able to integrate as proposed before the integration is recognized.

An experiment is conducted according to the topology in figure 13 and the settings in the section of Experiment Settings. The purpose of the experiment is to determine the appropriateness of the proposed theories and concepts. A crisp mathematical link cost calculation with Djikstra algorithm is chosen as the control model. The control model acts as a benchmark to test the effectiveness and efficiency of the proposed model. Number of

accepted connection and cell loss ratio are the metrics for measuring the performance of both models. These metrics are suitable for this experiment because the ideal route selection solution should maintain the network performance while maximizing the accepted connections. Please note that the link costs calculated by both models are relative link costs to each model and is not suitable to perform cross-model comparison.

The experiment result shows that the proposed model slightly outperforms the control model. The proposed model accepts one connection more than control model and the proposed model has lower overall average cell loss ratio. The experiment result proves that the theories and concepts integration proposed in chapter 3 is practical. The proposed model integrates load balancing, bandwidth packing, and congestion avoidance to achieve the ideal result. Load balancing with packing could balance the traffic load throughout the entire network while avoiding bandwidth fragmentation. Congestion avoidance predicts congestion and avoids passing through congested area.

The proposed model's artificial intelligence implementation is the main reason for its success. Fuzzy logic implemented in the proposed model allows several concepts to be integrated easily by transforming the complex numerical formulas into rules, a set of human understandable rules. The best example is the combination of load balancing with bandwidth packing. The integrated concept prevents the occurrence of bandwidth fragmentation while evenly distribute the network load, which leads to more connections to be accepted. Spiking neurons network with the ability to find all possible routes with each route's total link cost allows for a bigger solution space. Hence, spiking neuron network increases the possibility of accepting a new connection request.

As a conclusion, implementing artificial intelligence in dynamic routing problem could help solution model to become more flexible and adaptive. Fuzzy logic enables the solution

model to easily integrate a few concepts to handle different difficulties in dynamic routing problem. Spiking neurons network provides all alternatives route with each route's total link cost for selection. Combining fuzzy logic with spiking neuron network enables the solution model to be capable in finding a "most optimal" solution for the multi objectives problem, which is the dynamic routing problem in ATM.

Chapter 5

Conclusion

The main objective of this project is to propose an intelligent routing solution that increases the network performance in Asynchronous Transfer Mode (ATM) network for ABR traffic type. The proposed intelligent routing algorithm integrates neural network (spiking neuron network) with fuzzy logic (fuzzy link cost) to perform dynamic routing computation. The intelligent routing algorithm is responsible for planning network resources consumption and maintains network performance.

ATM network is one of the high speed networks which delivers its services with a guaranteed Quality of Service (QoS). ATM network uses a fixed size packet known as ATM cell for data transmission. ATM network adopts the concept of virtual path and virtual circuit on its routing mechanism. A pair of virtual path identifier and virtual circuit identifier (VPI/VCI) helps to uniquely identify each connection in the ATM network. The ABR (Available bit rate) data traffic is one of the non real time data traffic supported in ATM. ABR is introduced for bursty application that uses a reliable end-to-end protocol. Whenever an application requests for an ABR connection, the application specifies the required peak cell rate (PCR) and minimum cell rate (MCR) for the allocation of at least MCR capacity during the connection lifetime.

Although ATM network is a high-speed network, the increasing requirements in the application will cause network congestion in ATM network too. An effective dynamic routing algorithm must be able to adapt to changes in the network traffic and network state such as bursty traffic and network congestion. In network routing, obtaining topology

information and maintaining up-to-date information for each network node is very important. The gathered network information determines the effectiveness of the implemented routing algorithm.

The most popular and widely used routing algorithm is the least loaded routing. This routing algorithm is an implementation of the load balancing concept. Load balancing is a concept that tries to evenly distribute the network loads between heavily loaded links and under-utilized links. Load balancing is proven to be able to increase network performance significantly, especially under heavy or unbalanced network loads [39]. Distributing network load evenly is highly desirable for single rate traffic, but may not be the best strategy in multi rate environment. Modified least loaded routing is a dynamic routing algorithm that implements bandwidth packing with load balancing [20].

Bandwidth packing is a concept of packing similar or smaller bandwidth connections on high utilization links. This helps to save bandwidth on other low utilization links for higher bandwidth requirement connections. Implementing bandwidth packing helps to achieve two desired results [34]:

1. Minimize the problem of bandwidth fragmentation that could lead to higher utilization,
2. Improved fairness between narrowband and wideband connection by increasing the acceptance of wideband connection.

Load balancing and bandwidth packing concepts may contradict each other, but it is possible to incorporate both concepts into a single routing solution to cover each other's weaknesses.

Dynamic routing using artificial intelligence (AI) is getting more attention in network routing studies. The popular technologies are fuzzy logic and neural network because they can provide adaptive, model-free and real-time control. Fuzzy logic focuses on network controlling (CAC and UPC) and link cost calculation while neural network is mainly applied to find optimal routing path. Dynamic routing in ATM network is an instance of multi-objectives routing problem. Integration of neural network and fuzzy logic provides the answer for the complex multi-objectives routing problem in ATM.

The proposed solution in this thesis is uses fuzzy link cost and spiking neuron network to implement both concept of load balancing and bandwidth packing in ATM network dynamic routing problem. Fuzzy logic with the capability to simultaneously maximizing several selected objectives produces the fuzzy link cost for each link. The spiking neuron network then computes the “optimal” route with minimum cost using the calculated fuzzy link cost. The principle function of the proposed solution is to create a routing plan for the entire ATM network or ATM sub-network. The routing plan will try to balance the network load while trying to pack smaller connections at the used links with the guaranteed quality. The routing plan will influence every connection established by sending the source node a recommended DTL. The DTL generation process relies on the network state metrics feedback from each network node.

In the proposed model, the selected network state metrics for reference are current buffer size, buffer increase rate, bandwidth consumption and requested bandwidth. From these metrics, the fuzzy logic derives the link cost for each link based on its congestion level, acceptance of requested connection and load distribution. Congestion level is measured using buffer size and buffer increase rate as in indicators. In determining the requested acceptance level, fuzzy link cost uses the available bandwidth and requested bandwidth. For load distribution, load variance and the requested bandwidth are used. The computed

link cost will be passed to spiking neuron network for shortest path computation. The neurons in the spiking neuron network are arranged according to the ATM network topology. Each neuron in the neural network represents one network node and each synapse represents a link between ATM network nodes. A link cost is associated with a synapse. The link cost is considered as the propagation delay time (in millisecond) between the firing neuron and its paired neurons. The spiking neuron network determines the shortest path by going through several learning cycles. While going through learning cycles, the weight of each synapse is updated by the learning function and the correlation time decreases for every cycle. The correlation time is a parameter in the temporal correlation learning function that enables synapse's weight to increases or decreases variably according to the firing time of presynaptic neuron. After a number of cycles, some synapses' weight become larger while other synapses' weight decrease and disappear eventually. As a result, the survivor synapse connecting to each neuron will be the candidate route with the minimum link cost.

According to the experiment results, the proposed model outperforms the control model by accepting more connections while maintaining lower overall cell loss ratio. The experiment results prove that the proposed concept integration is practical. The proposed model's artificial intelligence implementation is the main reason for its success. Combining fuzzy logic with spiking neuron network enables the solution model to become capable of finding the "most optimal" for the multi objectives problem, which is the dynamic routing problem in ATM.

The proposed model proves that artificial intelligence implementation (using fuzzy link cost and spiking neuron network) realize concepts integration easily. The integrated concepts could cover each other's weaknesses and strengthen each other's strengths. The best examples of concept integration are load balancing and bandwidth packing: bandwidth

packing solves bandwidth fragmentation problem in load balancing and load balancing solves links under-utilization problem in packing bandwidth.

Future Work

In this study, only non real-time variable bit rate traffic with several connections is tested with the proposed model. The experiment returns a successful result with proposed model. However, an ATM network consists of various types of traffic and processes a lot of connections. The proposed model should be further tested with more connections and with heterogeneous traffics. In the handling of heterogeneous traffic, the ability to categorize and assign priority to each type of traffic must be taken into consideration. The other challenges that can be suited to the proposed model would be real-time traffic. Real-time traffic requires relatively faster response time and more accurate judgment on the route selection as performance degradation in real-time traffic is more noticeable than non real-time network traffic.

The proposed model is an initial study to be the planning unit on dynamic routing for the purpose of increasing the overall network performance. The ultimate objective shall evolve to become a planning unit for creation or destruction of virtual path and virtual circuits in ATM network. The execution of actual routing selection shall be handled by each route selection node.

References

[1] A. Tarraf, I. W. Habib, and T. N. Saadawi, “Characterization of Packetized Voice Traffic in ATM Networks Using Neural Networks Using Neural Networks”, *Proc. IEEE GLOBECOM ’93*.

[2] A. Tarraf, I. W. Habib, and T. N. Saadawi, “Congestion Control Mechanism for ATM Networks Using Neural Networks”, *Proc. ICC ’95*, Seattle, WA, June 1995.

[3] A. Tarraf, I. W. Habib and T. N. Saadawi, “Neural Networks for ATM Multimedia Traffic Prediction”, *Proc. Int’l WKSP. Appls. of Neural Networks to Telecommun. I*, Princeton, NI, Oct. 21-23, 1994, pp. 85-91.

[4] A. Pitsilides, Y. A. Sekereioglu, and G. Ramamurthy, “Fuzzy Backward Congestion Notification (FBCN) Congestion Control in Asynchronous Transfer Mode (ATM) Networks”, *Proc. GLOBECOM ’95*, Singapore, Nov. 1995, pp. 280-283.

[5] Aboelela, E. and Douligeris, C., “Routing in multimetric networks using a fuzzy link cost”, *Proc. Computers and Communications ’97, IEEE Symposium,* 1997, pp. 397-401.

[6] Atlasis, A. F., Baltatzis, E. D., Stassinopoulos, G. I. and Venieris, I., “A linear-based trunk reservation routing algorithm for ATM networks”, *Proc. LCN ’98*, 1998 pp.90-98.

[7] ATM Forum, "Private Network-Network Interface (PNNI) v.1.1 Specifications", ftp://ftp.atmforum.com/pub/approved-specs/af-pnni-0055.002.pdf, Apr. 2002.

[8] Bestavros, A. and Matta, I., “Load profiling for efficient route selection in multiclass networks”, *Proc. Int’l Conference ’97*, pp.183-190.

[9] Cidon, I., Rom, R. And Shavitt, Y., “Multi-path routing combined with resource reservation”, *Proc. IEEE INFOCOM ’97*, vol. 1, 1997, pp.92-100.

[10] Custodio, J. J., Tascon, M. and Dimitriadis, Y. A., “A new neuro-fuzzy system for efficient ATM traffic control”, *Artificial Neural Networks, ICANN 99,* vol. 6, pp. 964-969.

[11]D. M. Sala and K. J. Cios, “Self-organization in networks of spiking neurons”, *Australian J. Intell., Inform. Processing Syst.,* vol. 5, no. 3, pp.161 – 170, 1999.

[12] D.W. Glazer and C.Tropper, “A New Metric for Dynamic Routing Algorithms”, *IEEE Transactions on Communications*, Vol38, No 3, March 1990.

[13] E. S. Yu and Y. R. Chen, “Traffic Prediction Using Neural Networks”, *Proc. IEEE GLOBACOM ’93*, pp. 991-995.

[14] Felstaine, E. and Cohen, R., “On the distribution of routing computation in hierarchical ATM networks”, *IEEE/ACM Trans.*, vol. 7 no 6, Dec. 1999, pp. 906-916.

[15] Gang Feng and Zemin Liu, “ Dynamic routing algorithms in ATM networks”, *IEEE ISCAS ’98*, vol. 6, 1998, pp.498-501.

[16] Georgatsos, P. and Griffin, D., “A management system for load balancing through adaptive routing in multi-service ATM networks”, *IEEE INFOCOM ’96*, vol. 2, 1996, pp. 863-870.

[17] H. J. Fowler and W. E. Leland, “Local Area Network Traffic Characteristics, with implications for Broadband Network Congestion Management”, *IEEE ISAC*, vol. 9, no. 7, Sept. 1991, pp.1139-1149.

[18] H.Kung, T. Blackwell, and A.Chapman, “Credit-Based Flow Control for ATM Networks: Credit Update Protocol, Adaptive Credit Allocation, and Statistical Multiplexing”, *Proc. SIGCOMM ’94*, London, UK, Aug. 31-Sept. 2, 1994, pp. 101-114.

[19] H. R. Mehvar and T. Le-Ngoc, “A Neural Network Approach for Congestion Control in Packet Switch OBP, Satellite”, *Proc. ICC ’95*, Seattle, WA, June 1995.

[20] Hon-Wai Chu and Tsang, D.H.K., “Dynamic routing algorithms in VP-based ATM networks”, *IEEE GLOBECOM ’95*, vol. 2, 1995, pp. 1364 – 1368.

[21] Iwata, A. and Izmailov, R. and Sengupta, B., “Alternative routing methods for PNNI networks with partially disjoint paths”, *IEEE GLOBECOM ’98*, vol 1, 1998, pp.621-626.

[22] J.Wang, “A deterministic annealing neural network for convex programming,” *Neural Networks*, vol. 7, no. 4, pp. 629-641, 1994.

[23] J.Wang, “A recurrent neural network for solving the shortest path problem”, *Circuits and Systems I: Fundamental Theory and Applications, IEEE Trans.*, vol. 43, no. 6, June 1996, pp. 482-486.

[24] J.Wang, “Analysis and design of a recurrent neural network for linear programming,” *IEEE Trans. Circuits Syst. I*, vol. 40, pp. 613-618, Sept, 1993.

[25] K. F. Cheung et al., “Fuzzy logic Based ATM Policing”, *Proc. ICCS ’94*, 1994, pp.535-539.

[26] Li-Xin Wang, “A Course in Fuzzy Systems and Control*”*, Prentice-Hall Inc., 1997.

[27] Logothetis, D. and Veeraraghavan, M., “Delay sensitive routing in PNNI-based ATM networks”, *IEEE GLOBECOM 1998*, vol. 1, 1998, pp. 604-612.

[28] M. M. Ali and H. T. Nguyen, “A Neural Network Implementation of an Input Access Scheme in a High-Speed Packet Switch”, *IEEE Trans. Commun.*, vol. 42, no. 12, pp. 3189-3199.

[29] M. R. Garey and D. S. Johnson, “Computers and Interactability: A guide to the Theory of NP-Completeness”, *W. H. Freeman and Company*, NY, 1979.

[30] M. Steenstrup ed., "Routing in Communications Networks," *Prentice Hall*, Englewood Cliffs, New Jersey, 1995.

[31] Manju K. Ahuja, Kathleen M. Carley, Robert B.Dial, Kate A.Smith, “Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research”, INFORMS Journal on Computing, v.11 n.1, p.15-34, January 1999.

[32] O. Hu, D.W. Petr. and C. Brown, “ Self-tuning Fuzzy Traffic Rate Control for ATM networks”, *Proc. ICC ’96*, Dallas, TX, June 1996, pp. 424-428.

[33] R.G. Cheng and C. J. Chang, “Design of a Fuzzy Traffic Controller for ATM Networks”, *IEEE/ACM Trans. Networking*, vol. 4, no. 3, June 1996, pp. 460-469.

[34] S.Gupta. “Performance Modeling and Management of High-Speed Networks”. *PhD thesis*, University of Pennsylvania, 1993.

[35] S.Gupta, K. Ross and M. El Zarki, “On routing in ATM networks”, *IFIP Trans.* C-15, North Holland, pp. 229-239, 1993

[36] S. Haykin, “Neural Networks – A Comprehensive Foundation”, Second Edition, Prentice Hall, Upper Saddle River, NJ, 1999.

[37] Sala, D. M. and Cios, K. J., “Solving graph algorithms with networks of spiking neurons”, *Neural Networks, IEEE Trans.*, vol 10, no. 4, July 1999, pp. 953-957.

[38] Sima, J., Orponen, P. “A Computational Taxonomy and Survey of Neural Network Models”, Neural Computation, 2001, 12 (12), 2965-2989.

[39] Songnian Zhou. “Performance Studies of Dynamic Load Balancing in Distributed Systems”, *PhD thesis*, University of California Berkeley, 1987.

[40] Stallings, William, “High-Speed Networks: TCP/IP and ATM Design Principles”, Prentice Hall, Upper Saddle River, NJ, 1998.

[41] Steven D. Keahler, “Fuzzy Logic – An Introduction”, *The Newsletter of Seattle Robotics Society,* Issue March ‘98, http://www.seattlerobotics.org/encoder/mar98/index.html, 1998

[42] Trangmoe, Greg, “A comparative study of dynamic routing in circuit-switched networks”, Department of Electrical and Computer Engineering, University of Arizona, http://www.statslab.cam.ac.uk/~frank/DAR/dyroute.html, December 1995.

[43] Van Mieghem, P., “Routing in a hierarchical structure”, *ICATM-98.*, IEEE International Conference, 1998 , Page(s): 378 –384

[44] Vieira, F.C., Doria Neto, A.D., e Costa, J.A.F. (2003). “An Efficient Approach To The Travelling Salesman Problem Using Self-Organizing Maps”. International Journal of Neural Systems, Vol. 13, No. 2 (2003) 59-66

[45] W. Gerstner, “Time structure of the activity inneural network models”, *Phys. Rev. E*, vol 51, pp 738-758, 1995.

[46] W.Gerstner, and J.L. van Hemmen, “How to describe neuronal activity: spikes, rates, or assemblies?”, *Advances in Neural Information Processing Systems*, vol. 6, Morgan Kaufmann, San Mateo, pp. 463-470, 1994.

[47] W. Leland, “Window-Based Congestion Management In Broadband ATM Networks: The Performance Of Three Access-Control Policies, Proceedings”, *IEEE GLOBECOM ’89*, 1989, pp. 1794-1800.

[48] W. Maass. “Computing with spikes”. Special Issue on Foundation of Information Processing of TELEMATIK, 8(1): 32-36, 2002

[49] W. Maass, “Lower bounds for the computational power of networks of spiking neurons”, *Neural Computation*, vol. 8(1), pp. 1-40, 1996.

[50] W. Maass, “Networks of Spiking Neurons: The third generation of neural-network models,” *Neural Networks*, vol. 10, pp. 1695-1671, 1977.

[51] W. Maass, “On the computation complexity of networks of spiking neurons”, *Advances in Neural Information Processing Systems*, vol. 7, MIT Press (Cambridge), pp. 183-190, 1995.

[52] W. Maass, “Spiking Neuron Models: Single Neurons, Populations, Plasticity”, Cambridge University Press, Trumpington Street, Cambridge, United Kingdom, 2002.

[53] Y. Gong and I. Akyildiz, “Dynamic Traffic Control Using Feedback and Traffic Prediction in ATM networks”, *Proc. IEEE INFOCOM*, 1944, pp. 91-98.

[54] Y. K. Park, V. Cherkassky, and G. Lee, “Omega Network-based ATM Switch with Neural Network Controlled Bypass Queueing and Multiplexing”, IEEE ISAC, vol. 12, no. 9, Dec, 1994, pp. 1471-1430.

This page was intentionally left blank

APPENDICES

Appendix A

Detailed Results of Experiment

Appendix A

Detailed Results of Experiment

Experiment Software/Hardware

Software	Matlab Release 11 with fuzzy logic toolbox. The spiking neurons and some simulation components are built using the Matlab scripts.
Hardware	Pentium III, 256M RAM, 1 GB disk space

1. Control Model

Link Cost Calculation during request of connection 30 (Request Time: 0.5)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	19.3548	0	0	25
1-4	19.3548	0	0	25
2-3	19.3548	0	0	25
2-4	19.3548	0	0	25
3-5	19.3548	0	0	25
3-6	19.3548	0	0	25
4-5	19.3548	0	0	25
4-6	19.3548	0	0	25
5-7	19.3548	0	0	25
6-7	19.3548	0	0	25

Assigned Route: 1, 3, 5, 7, 11

Link Cost Calculation during request of connection 40 (Request Time: 1.0)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	67.7419	9.6774	0	29.8096
1-4	67.7419	0	0	25
2-3	87.0968	-9.6774	0	20.9664
2-4	87.0968	0	0	25
3-5	45.1613	9.6774	0.1	29.8596
3-6	64.5161	0	0.1	25.05
4-5	64.5161	-10.4167	0	20.6865
4-6	64.5161	0	0	25
5-7	22.5806	9.6774	1.7	30.6596
6-7	41.9355	-9.6774	0.1	21.0164

Assigned Route: 2, 3, 6, 7, 9

Link Cost Calculation during request of connection 50 (Request Time: 1.5)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	67.7419	3.2258	0	26.5101
1-4	67.7419	0	0	25
2-3	74.1935	-3.2258	0	23.5759
2-4	87.0968	0	0	25
3-5	45.1613	9.6774	0	29.8096
3-6	51.6129	6.4516	0.2	28.2115
4-5	64.5161	-10.4167	0	20.6865
4-6	64.5161	-6.4516	0	22.2329
5-7	22.5806	3.2258	3.2	28.1101
6-7	29.0323	-3.2258	0.3	23.7259

Assigned Route: 1, 4, 6, 7, 9

Link Cost Calculation during request of connection 60 (Request Time: 2.0)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	67.7419	3.2258	0	26.5101
1-4	67.7419	6.4516	0	28.1115
2-3	74.1935	-3.2258	0	23.5759
2-4	87.0968	-6.4516	0	22.2329
3-5	45.1613	9.6774	0	29.8096
3-6	51.6129	0	0	25
4-5	64.5161	-10.4167	0	20.6865
4-6	51.6129	0	0	25
5-7	22.5806	-3.2258	8.9	28.0259
6-7	16.129	3.2258	68.8	60.9101

Assigned Route: 2, 4, 5, 7, 9

Link Cost Calculation during request of connection 70 (Request Time: 2.5)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	74.1935	3.2258	0	26.5101
1-4	74.1935	0	0	25
2-3	80.6452	-3.2258	0	23.5759
2-4	80.6452	0	0	25
3-5	51.6129	3.2258	0.1	26.5601
3-6	58.0645	0	0.1	25.05
4-5	58.0645	-3.4722	0	23.4705
4-6	58.0645	0	0	25
5-7	16.129	3.2258	100	76.5101
6-7	22.5806	-3.2258	96.3	71.7259

Assigned Route: 1, 4, 6, 7, 10

Link Cost Calculation during request of connection 80 (Request Time: 3.0)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	74.1935	3.2258	0.1	26.5601
1-4	74.1935	3.2258	0.3	26.6601
2-3	80.6452	-3.2258	0	23.5759
2-4	80.6452	-3.2258	0	23.5759
3-5	51.6129	3.2258	0.1	26.5601
3-6	58.0645	-3.2258	0	23.5759
4-5	58.0645	-3.4722	0	23.4705
4-6	51.6129	3.2258	0.1	26.5601
5-7	16.129	0	99.9	74.95
6-7	16.129	0	100	75

Assigned Route: 2, 4, 5, 7, 10

Link Cost Calculation during request of connection 90 (Request Time: 3.5)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	74.1935	3.2258	0	26.5101
1-4	74.1935	0	0	25
2-3	80.6452	-3.2258	0	23.5759
2-4	74.1935	0	0	25
3-5	51.6129	0	0.2	25.1
3-6	58.0645	-3.2258	0	23.5759
4-5	51.6129	0	0	25
4-6	51.6129	3.2258	0.2	26.6101
5-7	9.6774	3.2258	100	76.5101
6-7	16.129	-3.2258	99.5	73.3259

Assigned Route: 1, 3, 6, 7, 10

Link Cost Calculation during request of connection 100 (Request Time: 4.0)				
Links	Network State Metrics			Cost
	Link Compatibility	Load Variance	Buffer Queue	
1-3	54.8387	6.4516	0	28.1115
1-4	54.8387	0	0	25
2-3	67.7419	-6.4516	0	22.2329
2-4	61.2903	0	0	25
3-5	38.7097	0	0	25
3-6	38.7097	0	0.1	25.05
4-5	38.7097	0	0.1	25.05
4-6	38.7097	0	0.1	25.05
5-7	-3.2258	0	99.9	0
6-7	-3.2258	0	100	0

Assigned Route: -

2. Proposed Model

Link Cost Calculation during request of connection 30 (Request Time: 0.5)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	19.3548	80.6452	0	0	0	33.3136
1-4	19.3548	80.6452	0	0	0	33.3136
2-3	19.3548	80.6452	0	0	0	33.3136
2-4	19.3548	80.6452	0	0	0	33.3136
3-5	19.3548	58.0645	0	-0.0002	0	33.3136
3-6	19.3548	58.0645	0	-0.0429	0	33.3136
4-5	19.3548	58.0645	0	0	0	33.3136
4-6	19.3548	58.0645	0	0	0	33.3136
5-7	19.3548	35.4839	0	-0.1845	0	34.5758
6-7	19.3548	35.4839	0	-0.0479	0	34.5758

Assigned Route: 1, 3, 5, 7, 11

Link Cost Calculation during request of connection 40 (Request Time: 1.0)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	12.9032	67.7419	0	-0.0013	9.6774	11.5772
1-4	12.9032	87.0968	0	0	0	31.9402
2-3	12.9032	87.0968	0	0	-9.6774	32.7932
2-4	12.9032	87.0968	0	0	0	31.9402
3-5	12.9032	45.1613	0.1	0.2093	9.6774	11.5772
3-6	12.9032	64.5161	0.1	0.2407	0	31.9402
4-5	12.9032	64.5161	0	0	-9.6774	32.7932
4-6	12.9032	64.5161	0	0	0	31.9402
5-7	12.9032	22.5806	1.7	-0.1083	9.6774	13.1138
6-7	12.9032	41.9355	0.1	0.2712	-9.6774	33.1349

Assigned Route: 2, 3, 5, 7, 9

Link Cost Calculation during request of connection 50 (Request Time: 1.5)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	12.9032	67.7419	0	-0.0218	3.2258	20.8227
1-4	12.9032	87.0968	0	0	0	31.9402
2-3	12.9032	74.1935	0	-0.0002	-3.2258	37.5005
2-4	12.9032	87.0968	0	0	0	31.9402
3-5	12.9032	32.25881	0	-0.7776	16.129	21.0945
3-6	12.9032	64.5161	0.1	0.3172	0	31.9402
4-5	12.9032	64.5161	0	0	-16.129	11.5772
4-6	12.9032	64.5161	0	0	0	31.9402
5-7	12.9032	9.6774	100	2.9456	16.129	43.2868
6-7	12.9032	41.9355	0.2	0.554	-16.129	11.72

Assigned Route: 1, 3, 6, 7, 9

Link Cost Calculation during request of connection 60 (Request Time: 2.0)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	12.9032	54.8387	0	-0.1243	9.6774	11.5772
1-4	12.9032	87.0968	0	0	0	31.9402
2-3	12.9032	74.1935	0	-0.0832	-9.6774	32.7932
2-4	12.9032	87.0968	0	0	0	31.9402
3-5	12.9032	32.2581	0.6	0.0173	16.129	21.0945
3-6	12.9032	51.6129	0	-0.0053	6.4516	11.5772
4-5	12.9032	64.5161	0	0	-16.129	11.5772
4-6	12.9032	64.5161	0	0	-6.4516	37.5005
5-7	12.9032	9.6774	100	0.2484	9.6774	43.2868
6-7	12.9032	29.0323	2.7	-2.011	-9.6774	35.5029

Assigned Route: 2, 3, 6, 7, 9

Link Cost Calculation during request of connection 70 (Request Time: 2.5)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	6.4516	61.2903	0	-0.0063	3.2258	20.8227
1-4	6.4516	93.5484	0	0	0	31.9402
2-3	6.4516	67.7419	0	-0.0444	-3.2258	39.0026
2-4	6.4516	93.5484	0	0	0	31.9402
3-5	6.4516	38.7097	0.2	-0.4984	16.129	20.2846
3-6	6.4516	45.1613	0.6	1.5685	12.9032	15.7375
4-5	6.4516	70.9677	0	0	-16.129	11.5772
4-6	6.4516	70.9677	0	0	-12.9032	22.2111
5-7	6.4516	16.129	100	0.194	3.2258	41.4599
6-7	6.4516	22.5806	68.5	3.2389	-3.2258	45.2725

Assigned Route: 1, 3, 6, 7, 10

Link Cost Calculation during request of connection 80 (Request Time: 3.0)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	6.4516	54.8387	0.6	1.9128	6.4516	11.5772
1-4	6.4516	93.5484	0	0	0	31.9402
2-3	6.4516	67.7419	0	-0.3252	-6.4516	39.5658
2-4	6.4516	93.5484	0	0	0	31.9402
3-5	6.4516	38.7097	0.1	-0.4284	16.129	20.2846
3-6	6.4516	38.7097	0.3	0.7088	16.129	20.2846
4-5	6.4516	70.9677	0	0	-16.129	11.5772
4-6	6.4516	70.9677	0	0	-16.129	11.5772
5-7	6.4516	16.129	100	0.0805	0	44.8211
6-7	6.4516	16.129	99.8	0.4461	0	44.8211

Assigned Route: 2, 4, 5, 7, 10

Link Cost Calculation during request of connection 90 (Request Time: 3.5)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	6.4516	54.8387	0.3	0.5203	6.4516	11.5772
1-4	6.4516	93.5484	0	0	-3.2258	39.0026
2-3	6.4516	67.7419	0	-0.1616	-6.4516	39.5658
2-4	6.4516	87.0968	0	0	3.2258	20.8227
3-5	6.4516	38.7097	0.1	-0.2076	12.9032	16.1843
3-6	6.4516	38.7097	0.4	0.385	16.129	20.2846
4-5	6.4516	64.5161	0	0	-12.9032	22.2111
4-6	6.4516	70.9677	0	0	-16.129	11.5772
5-7	6.4516	9.6774	100	0.2977	3.2258	43.2868
6-7	6.4516	16.129	99.7	1.7433	-3.2258	48.101

Assigned Route: 1, 3, 5, 7, 10

Link Cost Calculation during request of connection 100 (Request Time: 4.0)						
Links	Network State Metrics					Cost
	Requested Bandwidth	Link Compatibility	Buffer Queue	Average Dynamic	Load Variance	
1-3	19.3548	35.4839	0	-0.3481	9.6774	19.5827
1-4	19.3548	80.6452	0	0	-3.2258	35.8588
2-3	19.3548	54.8387	0	-0.0026	-9.6774	34.0829
2-4	19.3548	74.1935	0	0	3.2258	23.4624
3-5	19.3548	19.3548	0.1	-0.2731	16.129	23.6013
3-6	19.3548	25.8065	0.3	0.7796	16.129	22.1778
4-5	19.3548	51.6129	0	0	-16.129	11.5772
4-6	19.3548	58.0645	0	0	-16.129	11.5772
5-7	19.3548	-9.6774	100	0.1535	6.4516	52.1215
6-7	19.3548	3.2258	99.7	1.0234	-6.4516	50.0433

Assigned Route: 2, 4, 6, 7, 11