Fuzzy Control in Telecommunications

Hans Hellendoorn
Siemens AG, Dept. ZT SN 4, D-81730 Munich, hah@zfe.siemens.de

***Abstract*— Fuzzy logic is an excellent heuristic method to come to grips with the complexity problem in communication and computer networks. Four examples show the use of fuzzy logic to routing; first we show the use of explicitly available information in the form of routing tables, then we show the use of implicitly available information in the form of experience and heuristic knowledge in distributed networks. Then we show the use of fuzzy in Call Admission Control and Usage Parameter Control programs for broadband ISDN-networks.**

KEYWORDS: Fuzzy set theory, network routing, communication networks, CAC, UPC.

I. Introduction

The telecommunication market is one of the most rapidly rising markets, that offers an increasing number of services like video conferencing, telebanking, data exchange, video on demand, etc. Each of these services has its own requirements as there are loss tolerance, real time, costs, reliability, priority, etc.

Two highly connected questions are discussed in this paper: (1) Should a certain message be accepted by the network? and (2) How should a message be sent to the destination address based on (a) global information distributed by flooding certain routing tables or (b) local information extracted from input and output queues behavior? A third question that is discussed is how to deal with sending sources that do not behave according to the agreements made.

In Sections 2 and 3 we present the global and local fuzzy routing systems for computer networks. Sections 4 and 5 deal with CAC and UPC. Section 6 gives the conclusions.

II. A Global Fuzzy Routing System

Shortest path routing strategies are based on information about the network topology. According to a metric a length or weight is assigned to every link in the network. Then a shortest path algorithm (i.e., Dijkstra or Bellman-Ford) calculates the shortest path from a given source node to any other node in the network [5, 6]. In a distributed, adaptive routing algorithm every node in the network has to accomplish the following tasks: It has to collect/measure information about the network topology (there are numerous criteria depending on the network that influence the routing decision), it has to share the information with other nodes, usually by flooding, and it has to calculate the 'shortest' paths to all other nodes.

In all common shortest path routing strategies only one parameter is used as routing information. Different routing algorithms use different parameters, like transmission delay, number of hops, etc. But the routing decision is based exclusively on this one parameter, so the network traffic is optimized only considering, e.g., the delay time; other important criteria for a networking company like link costs are not taken into account.

The idea of our approach is to use more than only one parameter and to use a fuzzy system to obtain a crisp link length/link weight from a set of parameters. This link weight can easily be used as input for a shortest path algorithm. To show that this idea leads to a reasonable routing strategy we implemented a fuzzy link evaluator (FLE) with SieFuzzy, a fuzzy development tool from Siemens. We also implemented a network simulation model using the FLE with Opnet, a simulation system for telecommunication and computer networks.

As fuzzy systems can easily handle complex

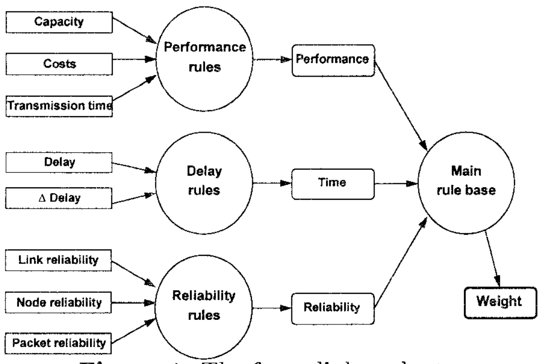

Figure 1: The fuzzy link evaluator.

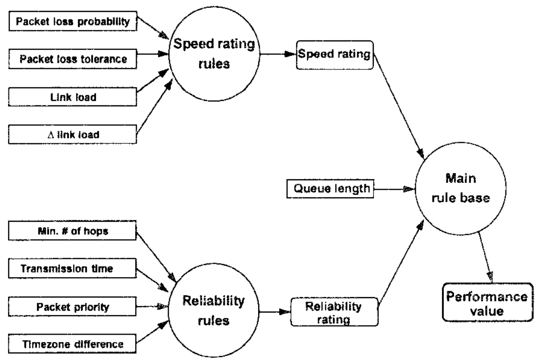

Figure 2: The local fuzzy routing algorithm.

situations we selected the following eight parameters as sensitive inputs for our fuzzy system: Link capacity, transmission cost, transmission time, transmission delay, change of transmission delay, link reliability, node reliability, and packet reliability. In the FLE these parameters are represented with linguistic variables and grouped into three categories to get a systematical design of the FLE. The first three parameters are used to represent the cost performance relation of the evaluated link. The next two represent the delay situation on the link and the last three are grouped to the link reliability, which may also be viewed as a trust in the link (Fig. 1). For these three groups (performance, timing, reliability) three rulebases are designed to compute three intermediate variables describing the three link characteristics of interest and serving as input for the main rulebase that computes the link evaluation. This FLE has been integrated into a simulation network to examine the behavior in a closed loop. With this simulation network it was tested how the fuzzy routing strategy reacted on a node failure, on 'bad' links with a high transmission time, rather high costs, and with a low reliability, and how it acted in comparison to a router without a fuzzy component.

The simulations show that using the FLE has an obvious influence on the routing decision. Especially the 'bad links' were better recognized by the fuzzy router than by the non-fuzzy router. We expected this result because the fuzzy router has better 'knowledge' about the network. This is one of the main advantages of our fuzzy routing algorithm. A fuzzy system can easily handle more than just one input parameter. The design and tuning of a FLE can easily be done by a network expert and changes to an existing system are quite simple because a well designed fuzzy system is easier to understand than huge sets of formulas. Furthermore, a fuzzy router can easily be adapted to changing conditions in a network.

III. A Local Fuzzy Routing System

In future there will be an increasing number of independent private network providers that will probably not be willing to distribute network information to neighboring networks. That means that global information about the network state, e.g., in the form of routing tables as in the former section is unavailable. In this case incoming and outgoing queue behavior of a local node can be used to estimate the network state.

First a simple router node model has been developed. All arriving packets are put into one waiting queue (FIFO). The server for this waiting queue is the routing algorithm itself. When a packet is served by the routing algorithm, it is put into the waiting queue attached to the outgoing link which is chosen by the routing algorithm. The servers of these waiting queues are given by low level network functions. We assume the servers to have an exponentially distributed service time for each packet. The queue lengths are bounded to a finite value. Therefore it may happen that a packet will find only full packet queues which may cause the loss of the packet. From this node model the input parameters of the network can be derived. The nine parameters, used in the local fuzzy routing system are: relative queue length, relative link load, load difference, time delay, minimum number of necessary hops, time zone, packet loss probability, packet loss tolerance, and packet priority.

The fuzzy algorithm contains three rulebases (Fig. 2). Rulebase 1 determines the reliability rating of a link, rulebase 2 the speed rating of a link. From these ratings rulebase 3 determines the performance value for the link. Here is a point where a network provider can influence the weighting of different criteria which finally determines the routing decision.

The analysis of the algorithm can be divided into a static and a dynamic part. The static analysis was done with SIEFUZZY which was also used to build the fuzzy decision module. This analysis just sets all parameters to certain values and traces what happens to the output in the case one parameter is changed. The dynamic analysis uses a simulation of a network to examine the effects when many parameters change continuously in a time interval. We used OPNET where we linked fuzzy modules to perform the routing inside of the nodes.

The dynamic analysis shows a sensitivity of the algorithm to the priority and the loss tolerance. The degree of sensitivity depends on the load conditions at the node. Furthermore, the dynamic analysis shows a decreasing packet loss rate.

IV. Call Admission Control

The CAC-algorithm is an instrument that decides whether an incoming call can be accepted or has to be rejected. The decision is based on a set of traffic descriptors that characterize each connection. These parameters together with a QoS (Quality of Service) specification are part of the traffic contract between the network and the user (the connection). QoS requirements may concern tolerance values of *cell delay*, *cell delay variation*, *cell loss ratio*, etc. If the network receives a new connection request, the CAC has to decide whether the network has enough resources left to provide the QoS-requirements to the new connection without affecting the QoS that have been guaranteed to the existing connections. Based on this estimation (there is no numerical analytical calculation method in the presence of statistical multiplexing) the call will be accepted or rejected.

Our proposed solution for a fuzzy CAC-system is based on an integrated fuzzy system performing, first, the estimation of the effective bandwidth (EffBw) that has to be reserved for a single connection, then the correction of the estimated effective bandwidth by measuring the network load and by estimating the suitability of the set of existing connections with regard to statistical multiplexing, and, finally, the comparison of the corrected effective bandwidth with the available link capacity.

As in ATM networks burstiness of connections is an essential criterion to define the effective bandwidth, the *burst factor* (resp. its reciprocal value) together with the *maximum burst size* (MBS) are used as inputs for the fuzzy rulebase. The burst factor is determined by the ratio of the parameters *sustainable cell rate* (SCR) and *peak cell rate* (PCR) as defined by the ATM forum in [2]. The fuzzy rulebase basically defines that connections are expected to perform bad with regard to statistical multiplexing if they have a large MBS and a SCR-PCR-ratio close to 1, i.e., these connections probably behave almost like a constant bit rate connection with an effective bandwidth close to PCR.

With regard to avoiding overload situations, which is a main goal of the CAC, this procedure performs quite well. However, network providers have to maximize the traffic that can be put onto a network without violating quality requirements. So, the CAC has to reach a maximum capacity utilization and therefore was enhanced by the following idea. If there is a sufficiently high number of accepted connections, which do not use the complete bandwidth that has been specified in the traffic contract and therefore do not need reservation of the effective bandwidth calculated above, then the difference between negotiated and used bandwidth can be used to correct the estimation of the effective bandwidth. Furthermore, one can increase the accuracy of the CAC by estimating the suitability of the accepted connections with regard to gain of statistical multiplexing. These two enhancements have been built in a fuzzy rule base that estimates how to correct the effective bandwidth based on the measured load difference, the number of established connections and the average ratio between SCR and PCR (Fig. 3).

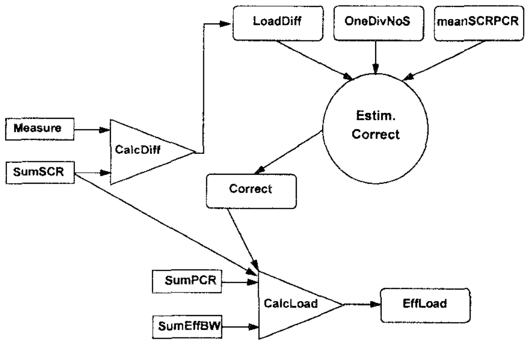

Figure 3: Fuzzy system to correct the estimated effective bandwidth

V. Usage Parameter Control

The UPC-algorithm supervises the established connections by checking and punishing violating connections. This is necessary because users of the network may not hold the agreed upon traffic parameters and therefore the CAC is insufficient to prevent congestion. Once a new connection has been accepted, the UPC is required to ensure that traffic submitted into the network does not exceed the parameters negotiated within the traffic contract. The traffic policing procedure UPC has to detect a source that does not keep the negotiated parameters very quickly. Otherwise connections that keep their negotiated parameters might be affected by delay or even by cell loss. The UPC can react using various actions for punishment like dropping, delaying, or marking violating cells. The UPC is located between the traffic source and the first node which manipulates the ATM-cell stream. Because the UPC works online, it must satisfy a strict real time demand. To recognize a violating cell stream some parameters have to be inspected. These parameters should be a subset of the CAC-controlled parameters.

The fuzzy-UPC shown in this paper is based on the parameters SCR, PCR, and MBS, similar to the CAC-algorithm. The number of ATM cells that want to enter the network, has to be limited to a specified amount reflecting the average cell rate. The critical point is the size of the time window which is used to enforce an average rate. To enable the fuzzy system to 'remember' the amount of cells which passed the UPC a token pool mechanism like in the leaky bucket model is used. Every cell that wants to pass the UPC to enter the network obtains a token—the token is erased when a cell uses it to pass on. The tokens are generated in a specified rate which depends on the SCR. Depending on the number of tokens in the token pool and the time distance between two cell arrivals the fuzzy system classifies every cell as *non-violating*, *partially violating*, and *violating*. Non-violating cells simply pass the UPC; partially violating cells are tagged so they can be eliminated in a node in case of congestion. Violating cells are eliminated immediately.

The peak cell rate is controlled strictly. If the time between two cell arrivals is lower than the minimum time allowed by the PCR, the cells are removed. The PCR is interpreted as an absolutely strict border which cannot be exceeded. The amount of tokens which can be stored in the token pool is directly connected to the degree of burstiness of the traffic. The burstiness is computed by a fuzzy system using the parameters SCR, PCR and MBS; the computed value specifies the amount of tokens which can be stored by the token pool.

The variables PCR, SCR and MBS are used as inputs in the first rule base to derive a general statement about the current burstiness. By now it is hard to obtain other input variables than the three mentioned above, the system could be greatly improved in the case more information about the burstiness of a particular message would be available. The burstiness together with the cell arrival time and the state of the bucket are used as inputs for a second rule base that decides the classification of the cell. Rules used are, e.g.,

IF Bucket_level is PM AND Cell Arrival is PB AND Burstiness is PM THEN Cell is rejected.
IF Bucket_level is PM AND Cell Arrival is PB AND Burstiness is PS THEN Cell is tagged.
IF Bucket_level is PS AND Cell Arrival is PB AND Burstiness is PVS THEN Cell is accepted.

VI. Conclusion and further work

The flexibility and easy management of a fuzzy system allows complex networking stategies. This is an interesting perspective—the use of multiple input parameters enables the router, the CAC- and the UPC-algorithms to consider application specific requirements, especially in the field of future broadband networks with a variety of services, like speech, video, file transfer.

In a future study we will take it for granted that another, conventional, CAC algorithm checks whether a message can be accepted or not. Although this might not allow full use of the available bandwidth, the fuzzy controller can in this case focus on higher interests of the networking company, e.g., which customer desires to send the message, what is the optimal mixture of messages that guarantees highest financial gain on the long term, should a message be rejected because it is to be highly expected that a more important message is coming in soon, etc.

Acknowledgement

I thank Christoph Thomas and Rudolf Seising for the excellent cooperation we had in implementing fuzzy notions in telecommunications applications. Furthermore and in particular, I want to thank the four eminent MSc students, Wolfgang Arnold, Werner Metternich, Matthias Nissel, and Andreas Weitzel for the great job they did, the lively discussions we had, and the cooperative way we could work together.

References

[1] Arnold, W., Hellendoorn, H., Seising, R., Thomas, C., Weitzel, A., "Network Routing with Fuzzy Logic - Two Case Studies," *Proc. of the EUFIT'95*, 1995, pp. 1735–1739.

[2] ATM-Forum, *Traffic Management Specification*, Version 4.0/5.0, 1995.

[3] Bonde, A.R., "A Comparative Study of Fuzzy Versus "Fixed" Thresholds for Robust Queue Management in Cell-Switching Networks," *IEEE/ACM Trans. on Networking*, **2**(1994)337–344.

[4] Dash, P.K., Liew, A.C., Rahman, S., "Peak Load Forecasting using a Fuzzy Neural Network," *Electric Power Systems Research*, **32**(1995)19–23.

[5] Dijkstra, E., "A Note on Two Problems in Connection with Graphs," *Num. Mathematics*, 1959.

[6] Ford, L.R., Fulkerson, D.R., *Flows in Networks*, Princeton, NJ, Princeton University Press, 1962.

[7] Hellendoorn, H., Metternich, W., Nissel, M., Seising, R., Thomas, C., *Traffic Management for Broadband Networks with Fuzzy Logic — Call Admission Control and Usage Parameter Control*, Internal paper Siemens.

[8] Khalfet, J., Chemouil, P., "Application of Fuzzy Control to Adaptive Traffic Routing in Telephone Networks," *Information and Decision Technologies*, **19**(1994)339–348.

[9] Ndousse, T.D., "Fuzzy Neural Control of Voice Cells in ATM Networks," *IEEE J. on Selected Areas in Communications*, **12**(1994)1488–1494.