Overload Detection and Control for Switching Systems using Fuzzy Rules

*Chung Hoon Rhee, **Byung Ho Rhee, and ***Sung Ho Cho

※This paper was supported in part by a 1997 Inter-University research fund from Hanyang University.

Abstract

In most switching systems, the processing unit is designed to work efficiently even at relatively high loads, but when the offered traffic exceeds a particular level, the rate of completed calls can fall drastically. A single call handled by the switching system consists of a sequence of events or messages that has to be processed by the control unit. The control unit is not only incapable of handling all of the offered calls, but also its call handling capability can drop as the offered load increases. The real time available for call processing is a critical resource that requires careful management. Therefore, the overloading of this resource must be detected by a subscriber in the form of a dial tone delay or an uncompleted call which is either blocked or mishandled. The subscriber may respond by either dialing prematurely or by re-attempting a call. This action can further escalate the processors load, which is spent for uncompleted calls. Unless a proper control is used, the switching system can finally break down. In this paper, we propose a fuzzy overload detection and control method for switching systems, which can be obtained by generating fuzzy rules via fuzzy aggregation networks. Simulation results involving a switching system is given.

I. Introduction

Most overload control strategies for switching systems consist of an overload indicator and a rejection procedure. If the overload indicator determines the system is in overload, the rejection procedure admits either no calls or a limited number. However, it is difficult to say whether the system is in an overload state and it is frequently not easy to find a parameter whose very-short term values provide an accurate indication of the state of the processor [1].

In this paper, we suggest a fuzzy overload detection and control method for switching systems, which can be obtained by generating fuzzy rules via fuzzy aggregation networks. In section 2, we describe an overload control model for a switching system and propose a fuzzy rule generation algorithm, which can estimate the membership functions, detect the redundant inputs, and generate the fuzzy rules from measured data. In section 3, we define the classes of the switching system load state and propose a fuzzy overload controller, which consists of an off-line learning and an on-line control function. To show the usefulness of the proposed fuzzy rule generation algorithm, the simulation of the load class detection for a small PABX switching system is carried out.

* Computational Vision and Fuzzy Systems Laboratory
** Soft Computing and Neural Networks Laboratory
*** Communications and Signal Processing Laboratory

Manuscript Received : October 15, 1998.

II. Overload Control and the Fuzzy Rule Generation Algorithm

The overload control of a switching system can be carried out by two functions : overload detection and control action. The overload detection procedure is a mapping of the monitored system parameters to the degree of the systems load state. The control action is an optimization procedure used to achieve the maximum engineered throughput and to determine the control input value against the degree of the systems load state. To implement the overload controller for a switching system, we need to select the appropriate monitored system parameters, classify the adequate degrees of the system load state, and seek the optimized control function. In doing so, we define the overload control problem for a switching system as follows. Let $X=(x_1, \ldots, x_n)$ be a set of system parameters, $Y=(y_1, \ldots, y_m)$ the degrees of the system load state, $U=(u_1, \ldots, u_l)$ the control input vector, $N(X, U)$ the system throughput, and N^* the engineered throughput. Then, the overload detection is to find a relationship between X and Y, such that rule $R_1 : X \rightarrow Y$, and the overload control is to find a relationship between (X, Y) and U, which can make the system throughput $N(X, U)$ approach N^*, such that rule $R_2 \mid N^* : (X, Y) \rightarrow U$.

However, it is not easy to find a relationship between the observed system parameters and the referenced load state, and a relationship between the system load state and

the control input, since the switching systems behavior is non-linear and time-delayed to the input load. Hence, it is difficult to find a set of system parameters which can reasonably discriminate the degree of the systems load state. Moreover, all the information that can be obtained is from measured data and knowledge of the switching operations are represented by vague descriptions. Therefore, we propose a fuzzy rule-generation method to find the set of system parameters and the relationships from measured data.

Properties (features) of the parameters related to the overload control can provide the necessary entities for proper rule generation. When features are not well defined, methods [2]-[5] for managing the uncertainty inherent in properties by means of hierarchical fuzzy aggregation networks have been discussed. In these hierarchical networks, the inputs to each node are the degree of satisfaction of each of the sub-criteria, and the output is the aggregated degree of satisfaction of the criterion. It has been shown that optimization procedures based on gradient descent can be used to determine the proper type of aggregation connective and parameters at each node, given only an approximate structure of the network and given a set of training data that represent the inputs at the bottom-most level and the desired outputs at the top-most level [2][3]. Also, it has been shown that such networks are capable of detecting certain types of redundant features in a decision-making problem.

If the attributes, properties, and relationships used in antecedent clauses of rules R_1 and R_2 in a switching system are interpreted as criteria, then one can model rule-based load status labeling also as a hierarchical network. In this paper, we use this idea and the redundancy detection capability of fuzzy aggregation networks to automatically generate fuzzy rules from measured (training) data.

2.1 Estimation of Membership Functions of Linguistic Labels

Let M be the number of outputs and K the number of inputs. The first step in rule generation is to generate the membership functions for the various linguistic labels that each input can take. As shown in [4], the member- ship functions may be estimated from smoothed histograms of the input values. Let $m_k^j(x_k)$ denote the smoothed histogram constructed from output j of input k. We now present a method to estimate the membership functions of the linguistic labels for an input.

A suitable parameterized function is chosen to model the membership function of a linguistic label. Let $h_k(x, \boldsymbol{p})$ denote the chosen parameterized function where $\boldsymbol{p} = (p_1, \ldots, p_n)$ is the parameter vector. Then each of the functions $m_k^j(x_k)$ can be approximated by a set $H_k^j = \{h_k^j(x, \boldsymbol{p}_i) | i = 1, \ldots, L_k^j\}$ of such parameterized functions, where L_k^j is the number of parameterized functions required for a reasonable approximation of $m_k^j(x_k)$.

To utilize a flexible function for modeling, we consider non-symmetric Gaussian function as suitable parameterized functions as

$$F(x) = \begin{cases} G(x, c, \sigma_1) & \text{if} \quad x < c \\ G(x, c, \sigma_2) & \text{if} \quad x \geq c, \end{cases} \tag{1}$$

where c is the mean value and $\sigma_1 (\sigma_2)$ is the left (right) standard deviation. If a membership function $m_k^j(x_k)$ consists of multiple peaks, we can model it as a sum of such L_k^j asymmetric functions as follows.

$$m_k^j(x_k) \approx \tilde{F}_k^j(x_k) = \sum_{i=1}^{L_k^j} a_{ki}^k F_{ki}^j(x_k), \tag{2}$$

where

$$F_{ki}^j(x) = \begin{cases} G(x_k, c_{ki}^j, \sigma_{1ki}^j) & \text{if} \quad x_k < c_{ki}^j \\ G(x_k, c_{ki}^j, \sigma_{2ki}^j) & \text{if} \quad x_k \geq c_{ki}^j \end{cases} \tag{3}$$

is the i^{th} parametrized function for feature x_k in class j. Here, $\boldsymbol{p}_{ki}^j = (a_{ki}^j, c_{ki}^j, \sigma_{1ki}^j, \sigma_{2ki}^j)$.

In order to approximate the smoothed histogram $m_k^j(x_k)$ for input x_k as a sum of parametrized functions, we could minimize the following objective function

$$J_k^j(\boldsymbol{p}_k^j) = \frac{1}{2} \sum_{i=1}^{L_k^j} \left[h_k^j(x_k, \boldsymbol{p}_{ki}^j) - m_k^j(x_k) \right]^2, \tag{4}$$

where $h_k^j(x_k, \boldsymbol{p}_{ki}^j)$ denotes the i^{th} parametrized function chosen to model the membership function $m_k^j(x_k)$ for input x_k in class j. We can now use a gradient descent method to estimate the parameter vector $\boldsymbol{p}_{ki}^j$ by the following update rule

$$\boldsymbol{p}_{ki}^{j(\text{new})} = \boldsymbol{p}_{ki}^{j(\text{old})} - \rho \frac{\partial J_k^j}{\partial \boldsymbol{p}_{ki}^j}, \tag{5}$$

where ρ is a positive learning constant. The parameter vector can be iteratively updated until there is little or no change in the parameter values. This occurs when the

partial derivatives of J_k^j with respect to each component of $\boldsymbol{p}_{ki}^j$ are approximately equal to zero (i.e., when the chosen approximation of parametrized functions $h_k^j(x_k, \boldsymbol{p}_{ki}^j)$ closely matches $m_k^j(x_k)$). If we use non-symmetric Gaussian functions as our choice for $h_k^j(x_k, \boldsymbol{p}_{ki}^j)$, the partial derivatives of J_k^j with respect to each component of $\boldsymbol{p}_{ki}^j = (a_{ki}^j, c_{ki}^j, \sigma_{1ki}^j, \sigma_{2ki}^j)$ are

and

$$\frac{\partial J_k^j}{\partial a_{ki}^j} = \left(F_k^j(x_k) - m_k^j(x_k)\right) F_{ki}^j(x_k), \tag{6}$$

$$\frac{\partial J_k^j}{\partial c_{ki}^j} = \begin{cases} \left(F_k^j(x_k) - m_k^j(x_k)\right) \dfrac{a_{ki}^j F_{ki}^j(x_k)(x_k - c_{ki}^j)}{(\sigma_{1ki}^j)^2} & \text{if } x_k < c_{ki}^j \\ \left(F_k^j(x_k) - m_k^j(x_k)\right) \dfrac{a_{ki}^j F_{ki}^j(x_k)(x_k - c_{ki}^j)}{(\sigma_{2ki}^j)^2} & \text{if } x_k \geq c_{ki}^j, \end{cases} \tag{7}$$

$$\frac{\partial J_k^j}{\partial \sigma_{lki}^j} = \left(F_k^j(x_k) - m_k^j(x_k)\right) \frac{a_{ki}^j F_{ki}^j(x_k)(x_k - c_{ki}^j)^2}{(\sigma_{lki}^j)^3} \tag{8}$$

for $l = 1, 2$.

For gradient descent methods, the choice of the initial values for the parameters is critical. Therefore, we can use a heuristic approach to obtain the initial parameters for the case of Gaussian functions [5].

2.2 Fuzzy Aggregation Networks for Rule Generation

The next step is to obtain a set of rules which possibly contain multiple antecedent clauses joined together either conjunctive or disjunctively. To achieve this, we use an approximate three-layer fuzzy aggregation network which shown in Fig. 1. From the figure, the bottom layer consists of K groups of nodes, with the k^{th} group consisting of L_k nodes. Each group corresponds to an input. The i^{th} node in the k^{th} group uses h_{ki} (the membership function of the i^{th} linguistic label for input k) as the activation function. The middle layer consists of K groups of M nodes each, where M is the number of outputs. The i^{th} node in group k in the bottom layer is connected to the j^{th} node in the corresponding grouping the middle layer if input k is considered non-redundant for output j and the support of h_{ki} has a non-empty intersection with the support of $m_k^j(x_k)$. Similarly, the j^{th} node of every group in the middle layer that has a connection from the bottom layer is connected to the j^{th} node of the top layer for $j = 1, \ldots, M$. All middle and top-layer nodes use a suitable fuzzy aggregation function (such as the generalized mean) as the activation function.

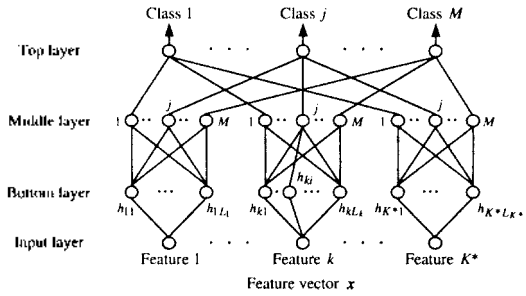


Figure 1. An approximate network structure for generating rules.

With this initial approximate aggregation network, training is initiated. The desired values in the training data are chosen to be 1 for the output from which the training data was extracted, and 0 for the remaining outputs. The learning is implemented using a modified gradient descent method as described in [2][3]. The nodes in the middle and top layers can represent either conjunctive or disjunctive nodes depending on the final values of the parameters of the aggregation function. Due to the constraints on the weights, some of the weights eventually become very small, indicating that the corresponding inputs are redundant. Thus, the training procedure has the ability to detect certain types of redundancies. When the training is complete, the resulting network can be interpreted as a set of fuzzy rules.

III. Fuzzy Overload Controller for a Switching System

For an effective overload controller, it must be able to provide control inputs in real-time, respond to the short and long-term overloads properly, and easily be constructed for various types of switching systems. Therefore, when selecting the control system parameters or structure of the overload controller, we consider the simplicity of the control algorithm and the design flexibility of the controller.

The number of offered calls and the occupancy rate of the processor are the main components of the monitored input vector X, since these can be monitored easily and can represent the overload symptoms prominently relative to the queue length or the response delay time for a call setup [6]. Coupling with the monitored inputs, we determine the number of admitted calls during the control unit time as the control input U, and select the total number of the completed calls as the system throughput N.

In order to achieve a certain degree of sensitivity to the short-term overload and to guarantee control stability, the concept of call holding time is introduced to select

an overload detection unit time and the degree of the system load state is classified into three categories. Next, the classification of the system load state is described [7].

3.1 Overload Detection and Classification

We define the different classes of overload using a throughput curve that is recommended by ITU-T as shown in Fig. 2. The engineered capacity N^* is designed to operate below the engineered processors occupancy ρ_m, and the engineered ratio of number of uncompleted and offered call, r_m. The tear-down point k_o on the slope of the offered load and throughput in Fig. 2 shows the starting point of deterioration of the system.

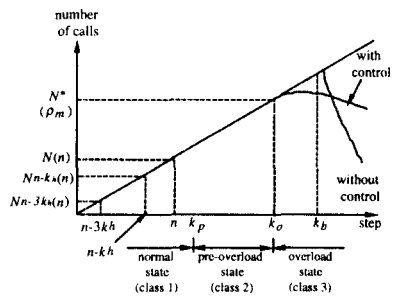


Figure 2. Classification regions of the load status.

It is necessary to detect this point prior to overload in order to obtain enough time to control the short and long-term overloads. Therefore, we define the load status of the system into three classes: normal state (class 1), pre-overload state (class 2), and overload state (class 3) as shown in Fig. 2. To describe the classes of the systems state, we consider the time it takes prior to overload. Since an overload starts after all the resources are occupied for the calls in service, it is reasonable to take the average call holding time t_h as the reference of the detection unit time t_d.

We now define the three classes of the systems load status. Let L be the set of input load trajectories possible to exist in a switching system such that $L = \{L_1(n), \ldots, L_z(n)\}$. A switching system characteristics can be presented by the set of state trajectories $S = \{S_1(n), \ldots, S_z(n)\}$, where a point of the state trajectory $S_i(n)$ can be defined by a q-dimensional vector, $S_i(k) = \{f_{i1}(k), \ldots, f_{iq}(k)\}$. The state features are selected among the switching system parameters or as a combination of ones.

Let $\{r_i(1), \ldots, r_i(n)\}$ be the sequence of the ratio of number of uncompleted and offered calls, and $\{\rho_i(1), \ldots, \rho_i(n)\}$ the processors occupancy generated from the input load trajectory $L_i(n)$, respectively. The set of system break-down points generated from $L_i(n)$, K_b^i, can be defined as $K_b^i = \{k\}$ such that $r_i(k+x), \ldots, r_i(k) > r_m$, $r_i(k-1) < r_m$ and $\rho_i(k+y), \ldots, \rho_i(k) > \rho_m$, $\rho_i(k-1) < \rho_m$. The set of overload recovery points generated from $L_i(n)$, K_r^i, can be defined as $K_r^i = \{k\}$ such that $r_i(k+x), \ldots, r_i(k) < r_m$, $r_i(k-1) > r_m$ and $\rho_i(k+y), \ldots, \rho_i(k) < \rho_m$, $\rho_i(k-1) > \rho_m$, where $x, y \geq k_h$. Let k_h denote the call holding step time (i.e., $k_h = t_h/t_d$). Then, the set of overload start points generated from $L_i(n)$, K_o^i, can be defined as $K_o^i = \{k\}$ such that $k_o^i = k_b^i - k_h$ and $k_b^i \in K_b^i$, and the set of pre-overload start points generated from $L_i(n)$, K_p^i, can be defined as $K_p^i = \{k\}$ such that $k_p^i = k_o^i - 3k_h$ and $k_o^i \in K_o^i$. Then, the load status of a switching system can be classified as an overload state class C_o if

$$C_o = \bigcup_{i=1}^{z} C_o^i, \quad \text{where } C_o^i = \{S_i(k) \mid K_o^i \leq k \leq K_r^i\}$$

and pre-overload state class C_p if

$$C_p = \bigcup_{i=1}^{z} C_p^i, \quad \text{where } C_p^i = \{S_i(k) \mid K_p^i \leq k \leq K_o^i\}$$

and normal state class C_n if

$$C_n = S - C_o - C_p.$$

Using the above definitions, we can formulate the overload detection process as a classification problem.

3.2 Fuzzy Load Estimator and Controller

The functional block diagram of the fuzzy overload controller is shown in Fig. 3. The fuzzy overload controller consists of three basic components: a system identification block, control tuning block, and fuzzy control block. The system identification and control tuning blocks are located in the simulation model and are operated in the off-line mode. The fuzzy control block is coupled with the real switching system and is designed to react to the short-term overload.

Before implementing the proposed fuzzy overload controller, it is required to prepare two sets of training data for estimating the rules R_1 and R_2. For identifying the system overload characteristics, the pairs (X, Y) are collected from the simulation model or the real system. We

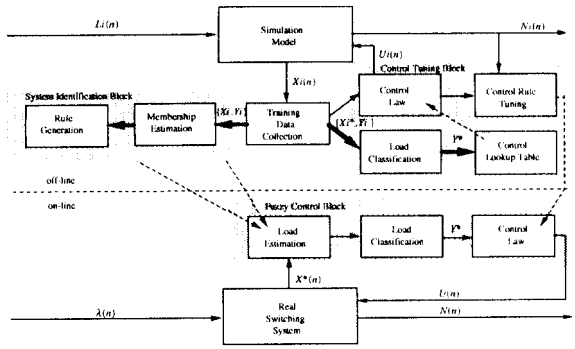


Figure 3. A functional block diagram for the fuzzy overload controller.

choose the number of the offered calls at control unit time $\lambda(n)$, the accumulated number of the offered calls for a mean call holding time $\lambda_{n,kh}(n)$, and the accumulated number of the offered calls for three times the mean call holding time $\lambda_{n,3kh}(n)$ as x_1, x_2, and x_3, respectively. The occupancy rate of the processor at control unit time $\rho(n)$, the accumulated occupancy rate of the processor for a mean call holding time $\rho_{n,kh}(n)$, and the accumulated occupancy rate of the processor for three times the mean call holding time $\lambda_{n,3kh}(n)$ are selected as x_4, x_5, and x_6, respectively. The class of the load state is selected as Y. For tuning the control rule R_2, the pairs of (X, Y, U, N) are collected. We choose the number of allowed calls for next control unit time as U and the number of completed calls as N.

The fuzzy overload controller is organized as follows:

Step 1: By inserting the standard overload input patterns to a simulation model or a real system, collect the $(x_1, \ldots, x_6, Y)$ training data. Remove the redundant input components (if any) using redundancy detection methods [4]. Estimate the membership functions of the linguistic labels for the non-redundant inputs X^*. Generate the fuzzy rule R_1 between X and Y, using the fuzzy aggregation network.

Step 2: As a control law, perform a_no-control_action for the normal load state and reject the offered calls for the overload state. With the results from step 1, subdivide the pre-overload state into q levels and make various control lookup tables which consist of algorithms for reducing the rate of the allowable offered call according to the pre-overload levels. Collect the (X^*, Y, U, N) data by utilizing the standard overload input patterns and control lookup tables. Find the control lookup table that maximizes the system throughput.

Step 3: Collect the X^* data for a control unit time and determine the systems overload state by using the fuzzy rule R_1 and the maximum defuzzification scheme. Apply the control input that is selected from the control lookup table.

Step 4: Save the $(x_1, \ldots, x_6, Y)$ data pairs collected at sample time t_s. Update the control lookup table using steps 1 and 2 in the off-line mode.

IV. Simulation Results

To show the effectiveness of the proposed fuzzy rule generation algorithm, we carry out some simulations using a simulation model called EMS PABX [8]. All simulation were performed with the Distributed Processor System Simulator (DPSS) as described in more detail in [6]. In the model, t_d and t_h are chosen to be 10 and 90 seconds, respectively. Also, $r_m = 0.05$ and $\rho_m = 0.75$. The training data was collected by simulating 3 types of the input load trajectories that can cause an overload to the system: $I_{-1}(n) = N_c \rho_m n$, $I_{-2}(n) = N_c(1.0 - \rho_m) n/2(k_h) + N_c(\rho_m - 0.1)$, and $I_{-3}(n) = 2N_c(1.0 - \rho_m) n/k_h + N_c(\rho_m - 0.1)$.

The described simulation may be considered as a 3-output/6-input classification problem. We extracted 50 samples from each output for the training data and each of the six inputs was mapped into the interval [0,1]. Inputs 1, 4, and 5 were eliminated easily using redundancy detection methods [4].

Fig. 4(a) shows the smoothed histograms for one of the three non-redundant inputs and Fig. 4(b) shows the membership functions of the resulting linguistic labels generated by fitting non-symmetric Gaussian to the smoothed histograms. The resulting approximate network for rule generation and final reduced network after training are shown in Fig. 5. Table 1 shows the final weights and the parameter p values. The rules obtained from the final network are listed below.

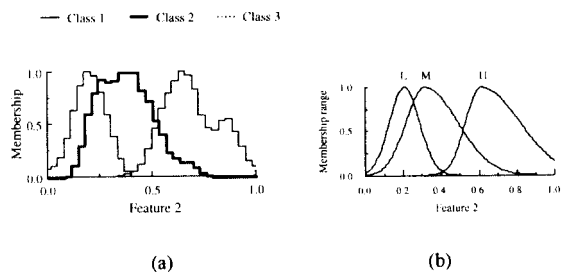
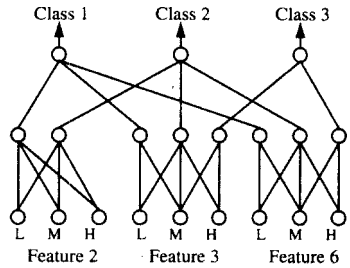
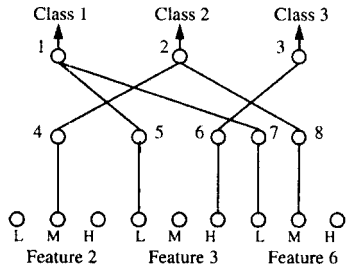


Figure 4. (a) Smoothed histograms and (b) Gaussian fitted linguistic labels for feature 2.



(a)



(b)

Figure 5. (a) Approximate network structure for generating rules (b) Reduced network after training.

Table 1. Values of weights and parameter p for the reduced network.

node	1	2	3	4	5	6	7	8
weights	0.476 0.524	0.274 0.726	1.000	1.000	1.000	1.000	1.000	1.000
p	-6.021	-0.615	1.018	0.922	0.988	0.982	1.006	-1.121

Rule 1 : IF **the accumulated number of offered calls** ($3k_h$ interval) is **LOW** AND
accumulated processor occupancy ($3k_h$ interval) is **LOW**
THEN the class is **Normal State.**

Rule 2 : IF **the accumulated number of offered calls** (k_h interval) is **MEDIUM** AND
accumulated processor occupancy ($3k_h$ interval) is **MEDIUM**
THEN the class is **Pre-overload State.**

Rule 3 : IF **the accumulated number of offered calls** ($3k_h$ interval) is **HIGH**
THEN the class is **Overload State.**

The outputs of the rule generation network for each sample was defuzzified using the maximum-membership defuzzification scheme. The rate of correct classification resulted in 92%.

V. Conclusions

The problem of overload control for a switching system can be defined as finding a relationship between the monitored system parameters and the degree of the system load state, and also finding a relationship between the system load status and optimal control input. Due to non-linearity and time-varying switching systems internal behavior, it is difficult to find parameters to express the overload status prominently. Also, it is difficult to extract overload detection and optimal control rules only with measured data. To provide a flexible overload controller, we proposed a fuzzy rule generation algorithm and a fuzzy overload controller. We classified the switching system load state into normal, pre-overload, and overload states and subdivided the pre-overload state into q levels. Fuzzy relations between monitored system parameters and the degree of the system load state were extracted by using estimated membership functions and fuzzy aggregation networks. The fuzzy overload controller and fuzzy control block which operates in two phases, was suggested. Simulation results show that the proposed fuzzy rule generation algorithm is effective in the overload detection and control in switching systems.

References

1. S.S. Joo, "General approach to overload control for the SPC switching system," *Proc. ICCS'90*, vol.2, pp. 688-691, Nov. 1990.
2. R. Krishnapuram and J. Lee, "Fuzzy-connective-based hierarchical aggregation networks for decision making," *Fuzzy Sets Syst.*, vol. 46, no. 1, pp. 11-27, Feb. 1992.
3. R. Krishnapuram and J. Lee, "Fuzzy-set-based hierarchical networks for information fusion in computer vision," *The Journal of Neural Networks*, vol. 5, no. 2, pp. 335-350, March 1992.
4. R. Krishnapuram and F. C.-H. Rhee, "Compact fuzzy rule base generation methods for computer vision," *Proc. 2nd IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE'93)*, San Francisco, CA, vol. II, pp. 809-814, March 1993.
5. F. C.-H. Rhee and R. Krishnapuram, "Fuzzy rule generation methods for high-level computer vision," *Fuzzy Sets Syst.*, vol. 60, pp. 245-258, Dec. 1993.
6. S.S. Joo, "Overload control algorithms for the stored program control switching system," Ph.D. Thesis, Seoul National Univ., Feb. 1989.
7. S. S. Joo and F. C.-H. Rhee, "A Fuzzy rule-based overload detection method for switching systems," *Proc. of IFSA'95*, San Paulo, Brazil, vol. II, pp. 691-694, July 1995.
8. C. H. Yim and H. L. Hartmann, "Throughput behavior of switching systems under heavy load conditions," *Proc. ITC-11*, paper 5.1B-5, 1985.

▲Chung-Hoon Rhee

Chung-Hoon Rhee was born March 3, 1963, in Seoul, Korea. He received the B.S. degree in electrical engineering from the University of Southern California (USC), Los Angeles, in 1985. He attended the University of Missouri, Columbia, from 1985 to 1987 and from 1990 to 1993, where he received the M.S. and Ph.D. degrees in electrical engineering respectively.

From 1990 to 1993, he was a research assistant in the Department of Electrical and Computer Engineering, University of Missouri, Columbia, where his research interests included applications of computer vision, pattern recognition, neural networks, and fuzzy set theory. From 1994 to 1995, he was a Senior Member of the Engineering Staff in the High-speed Network Access Section at the Electronics and Telecommunications Research Institute (ETRI), Taejon, Korea, where his work involved applications of group communications. Since September 1995, he has been a faculty member in the Department of Electronic Engineering at Hanyang University, Ansan, Korea, where he is currently an Assistant Professor. His current research interests include applications of computer vision, pattern recognition, and all aspects of computational intelligence.

Dr. Rhee was a co-chair for the Image Processing and Pattern Recognition area for the 1997 IEEE International Conference on Neural Networks held in Houston, TX, June 8-12. He is a member of the IEEE, the Korea Institute of Telematics and Electronics, the Korea Fuzzy Logic and Intelligent Systems Society, and the Acoustical Society of Korea.

▲Byung Ho Rhee

Byung Ho Rhee received the B.E. and M.E. degrees in electronic engineering from Hanyang University, Seoul, Korea, in 1975 and 1977, respectively, and the Ph.D. degree in electrical and electronic engineering from Chiba National University, Japan, in 1993.

From 1980 to 1981, he was a Member of the Technical Staff at the Electronics and Telecommunications Research Institute (ETRI), Seoul, Korea. From 1990 to 1991, he was a researcher at the Toshiba Research Institute. In 1981, he joined the Department of Electronic Engineering at Hanyang University, Ansan, Korea, where he is currently a Full Professor. His current research interests include software engineering, data communications, and neural networks.

▲Sung Ho Cho

Sung Ho Cho received the B.E. degree in electronic engineering from Hanyang University, Seoul, Korea, in 1982, the M.S. degree in electrical and computer engineering from the University of Iowa, Iowa City, USA, in 1984, and the Ph.D. degree in electrical engineering from the University of Utah, Salt Lake City, USA, in 1989.

From August 1989 to August 1992, he was a Senior Member of the Technical Staff at the Electronics and Telecommunications Research Institute (ETRI), Taejon, Korea. In September 1992, he joined the Department of Electronic Engineering at Hanyang University, Ansan, Korea, where he is currently an Associate Professor. His current research interests include digital communications, wireless mobile communications, digital signal processing and its applications, adaptive filtering, and stochastic processes. Dr. Cho is a member of Eta Kappa Nu and Tau Beta Pi.