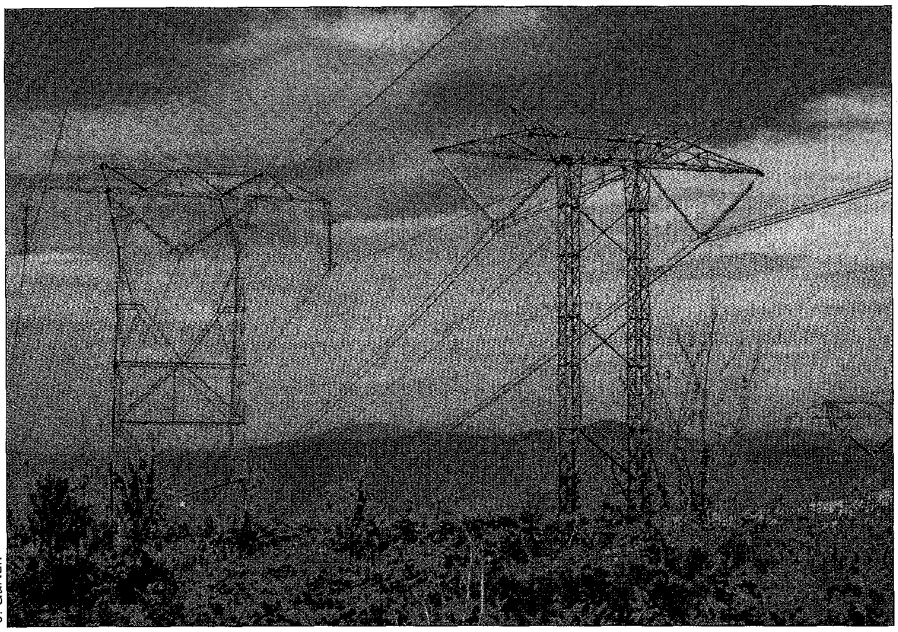
Detect and Classify Faults Using Neural Nets

Mladen Kezunovic and Igor Rikalo**



J. GaNun

A neural network trained to recognize patterns of transmission line faults is incorporated in a PC-based system that analyzes data files from substation digital fault recorders

The analysis of transmission line faults is essential to the proper performance of the power system. It is required if protective relays are to take the appropriate action and in monitoring the performance of relays, circuit breakers, and other protective and control elements. The detection and classification of transmission line faults is a fundamental component of such fault analysis.

Another application of fault analysis is in software packages for automated analysis of digital fault recorder (DFR) files. Recently, such a package, called DFR Assistant, was developed for substation applications. This program can be installed locally in a substation, in which case it is connected directly to the DFR via a high speed parallel link, or it can be installed at a central station, in which case it can be configured to automatically analyze events coming from all DFRs.

The substation installation of this software requires a stand-alone PC connected to a DFR. As soon as the DFR records a disturbance, data is automatically transferred to the PC and the fault analysis software determines if a fault has occurred and, if so, the type of fault. This conclusion is used to check if the relay operation is correct or not.

If installed at a central location, master station communication software is used to retrieve events from DFRs over a modem connection. Once the new event is retrieved, it is analyzed and, if appropriate, archived where it can be accessed by protection engineers for future examination.

Existing Fault Analysis Techniques

One of the most common techniques utilized for fault analysis is one based on the symmetrical components theory. This technique has been used for over 50 years in various protective relay applications. This technique requires computation of symmetrical component phasors, resulting in positive, negative, and zero sequence phasors. The computation of phasors requires appropriate processing considerations when used in digital relays and other computer-based applications.

The most important considerations in computing power system phasors are the sampling rate, antialiasing filters, and data window. The sampling rate is determined by the method used for phasor computation, such as Fourier transform. The antialiasing filters are used to band limit the frequency spectrum of the input

* Texas A&M University, Test Laboratories International, Inc.

ISSN 0895-0156/96/$5.00©1996 IEEE

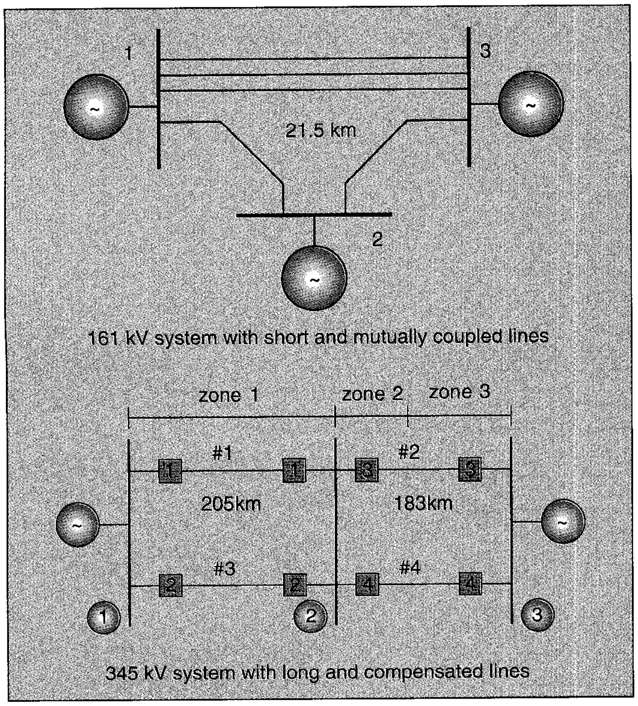


Figure 1. EMTP models used for simulations

current and voltage to meet the sampling theorem which states that the sampling frequency should be at least twice the highest frequency in the spectrum. The data window consideration relates to the number of samples required to compute a phasor. The most common data window is one cycle (16.67 ms for 60 Hz signal). However, some half-cycle techniques are also used.

Once symmetrical component phasors are calculated, a known theory of fault analysis is applied to determine fault occurrence and fault type.

New Approaches

Another approach that can be used for fault detection and classification is to utilize samples of currents and voltages directly without computation of phasors and related symmetrical components. There is no need to perform extensive filtering to obtain phasors. Instead, transient waveform data can be utilized directly to perform the required processing. In addition, the data window can be quite short and does not need to satisfy particular rules present for the phasor calculation. This new approach is based on the use of neural networks.

A *neural network* is a parallel, distributed, information processing structure consisting of processing elements (which can possess a local memory and carry out localized information processing operations) interconnected together with unidirectional signal channels called connections. Each processing element has a single output connection that branches ("fans out") into as many collateral connections as desired (each carrying the same signal, the processing element output signal). The processing element output signal can be of any mathematical type desired. All of the processing that goes on within each processing element must be completely local; i.e., it must depend only upon the current values of the input signal arriving at the processing element via impinging connections and upon values stored in the processing element's local memory. The key elements of most neural-net descriptions are distributed representation, the local operations, and nonlinear processing. Neural networks are primarily used in situations in which only a few decisions are required from a massive amount of data and situations in which a complex nonlinear mapping must be learned. Main applications of present-day neural-network computing include:

- Functional approximation
- Clustering
- Data compression
- Optimization
- Topological mapping.

Neural-Net Structure

For a clear understanding of potential benefits achieved with the new neural network approach, it is important to emphasize specific properties and characteristics of both the power system application and the neural-net implementation.

The transmission line fault analysis application embodies several data processing properties. Data acquisition is aimed at collecting samples of analog quantities (voltages and currents) from secondaries of instrument transformers, and status information (contacts) from circuit breakers, switches, and protection relays. Samples of the analog quantities must be processed simultaneously for all the voltages and currents on a transmission line. This facilitates timely determination and comparison of the signal parameters and time sequences of contact changes.

The process of comparing requires an easy interfacing between the signal and logic processing. The final outcome of the fault analysis can be obtained with high selectivity and speed, since all decisions are based on instantaneous changes of the signal parameters and the corresponding sequence of events.

The fault analysis application, as defined in the context of this article, requires that fault detection and classification are determined in the following manner:

- The outcome of the processing must be presented in a symbolic form (class names), since the detection and classification results of the neural-net computation may be further utilized in a rule-based expert system.
- Neural-net training must be quite efficient and straightforward, since the fault analysis application requires a fast and simple procedure for adapting to the changing power network conditions.

Fault detection and classification is defined as a multiclass problem. The eleven types of faults (a-g, b-g, c-g, a-b, b-c, c-a, ab-g, bc-g, ca-g, abc, abc-g) and the no-fault situation produce a twelve-class classification problem.

A literature search indicates that most of the neural-net implementations for fault detection and classification are based on multilayer, feed-forward nets. In this case, the application is considered to be a mapping problem. Supervised learning can be used where sets of associated input/output pairs are presented to a neural net that then *learns* a model of that process. However, the training process of multilayer networks is computationally demanding, and, in some instances, tens of thousands of iterations are needed to achieve convergence. Such performance may not be suitable for fast fault detection and classification. Since our problem is a classification problem, where only discrete labeling of classes is needed, the use of feed-forward networks may not be fully justified under stringent processing time requirements.

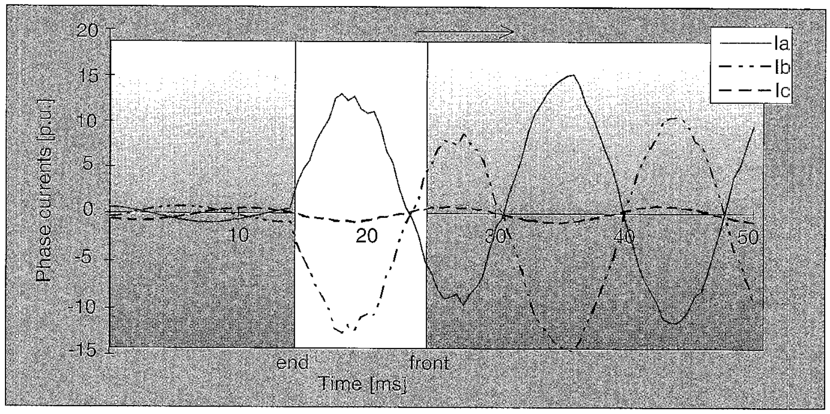


Figure 2. Sliding data window input to the neural net

Another possible approach for the neural-net application to our problem is to exploit data self-organization obtained through the use of unsupervised learning. After the learning (cognition phase), the user defines or labels clusters according to some criterion. The neural net is then ready for the classification task (recognition phase). Therefore, the concept of data self-organization through the use of unsupervised learning is valuable for discovering how an ensemble of patterns is distributed in the pattern space.

To overcome the mentioned limitations of the multilayer, feed-forward networks, and to take advantage of the suitability of self-organizing networks to perform a classification through the clustering process, a new neural-network approach has been developed and applied in our study. It incorporates advantages of both supervised and unsupervised training procedures and yet meets the requirements presented earlier. The proposed method utilizes the concept of supervised clustering, which demonstrates the following important properties:

- The number of iterations in the learning process is greatly reduced using unsupervised learning with a supervised class membership inheritance process.
- The training is far less complex than in standard supervised learning.
- Combining symbolic and numeric data is readily available.

Neural Network Implementation

The following steps are needed in order to use any existing neural net structure:

- Select the neural net algorithm most suitable for a given application
- Define a detailed training data set that represents cases the neural net needs to learn
- Train the neural net
- Test the neural net using a test data set until satisfied with its performance.

The neural-net algorithm used for this application embodies the ISODATA clustering algorithm, which is well-known in classical pattern recognition. This type of neural net assumes no teaching and performs unsupervised learning. The process performs comparison of a given input with previously encountered patterns. If the input is similar to any of the patterns, it will be placed in the same category. If the input is not similar to any of the previously presented patterns, a new category will be assigned. Category proliferation is controlled by the confidence measure, called *vigilance.* It calibrates how well an exemplar needs to match the prototype that it selects in order for the corresponding category to be chosen. If vigilance is low, even poor matches are accepted. Many different exemplars can then be incorporated into one category, so compression and generalization by that category are high. If vigilance is high, then even good matches may be rejected, and hypothesis testing may be initiated to select a new category. In this case, few exemplars activate the same category, so compression and generalization are low. A very high vigilance can select a unique category for a rare event that predicts an outcome different from that on any of the similar exemplars that surround it. In this case, the prototype of the category learns the unique exemplar that the category represents.

The training data set is generated with extensive EMTP simulation using models of actual power systems. The selected models of power systems contained both short and long transmission lines and mutually coupled lines. The one-line diagrams of the selected EMTP models are shown in Figure 1.

Thousands of cases were run by changing fault type, inception angle, fault resistance, and fault location. Phase currents and voltages were recorded for each case. All of the different cases were then divided into two sets, one to be used for neural-net training and the other

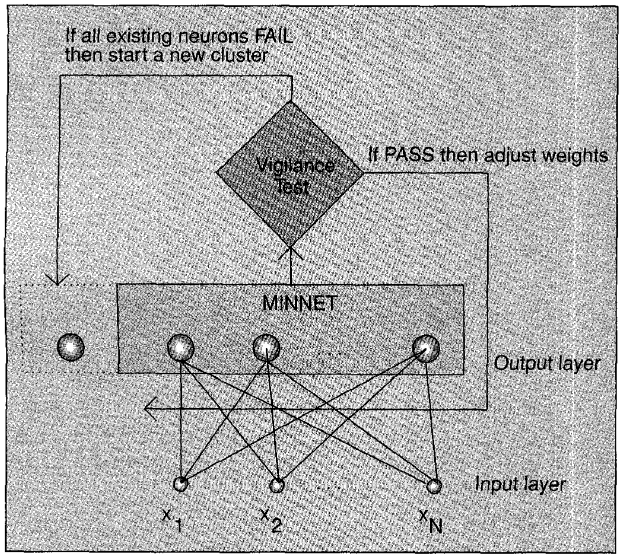


Figure 3. Neural net training procedure (clustering)

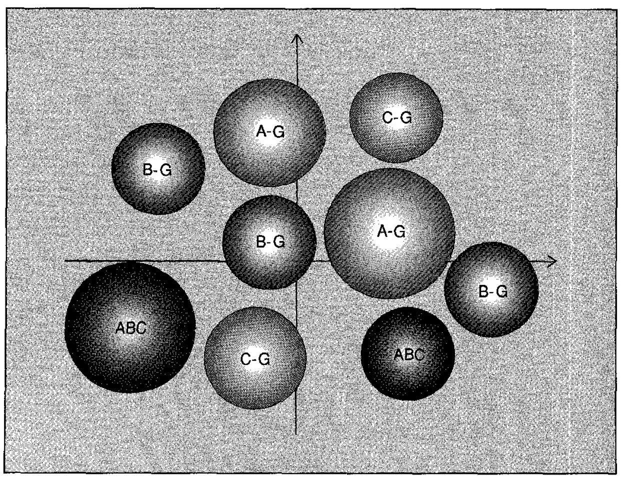


Figure 4. Outcome of neural net training

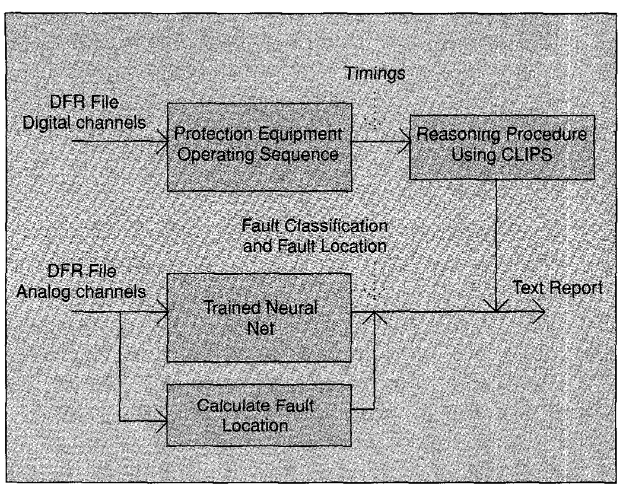


Figure 5. Hybrid system utilizing neural net and expert system

for testing. The input data set for the neural net is organized in the form of a sliding data window with a fixed window length of 1 cycle of the fundamental waveform (16.67 ms of data). The sampling frequency was 2 kHz. This corresponds to the 33 samples per cycle. The neural net was fed with three different signals (three phase currents) which gives total of 99 input neurons. The current samples are arranged in the input vector according to the following:

$$[I_{a1}\, I_{b1}\, I_{c1}\, I_{a2} I_{b2}\, I_{c2} \ldots\, I_{a32}\, I_{b32}\, I_{c32}\, I_{a33}\, I_{b33}\, I_{c33}]$$

An example of the sliding data window technique is shown in Figure 2, where the arrow points to the sliding direction. Sliding motion is obtained by putting every new sample at the front of the window and removing the first sample from the end of the window.

The neural net configuration is presented in Figure 3. It generates clusters by itself, if such clusters can be identified in the input data. Essentially the network *follows the leader* after it associates the first cluster with the first input vector received.

The algorithm works as follows.

- First, an input vector (training pattern) becomes, by default, a member of the first cluster defined by the neural net.
- Second, the training pattern is matched against the existing cluster. If the Euclidean distance between the input vector and the center of the cluster is less than a threshold (vigilance test), the matching test is passed, the pattern is added to the cluster, and the cluster center adjusted accordingly. If the Euclidean distance is greater than a threshold, the matching test fails, and a new cluster is created with one member (second training pattern).
- This procedure is reiterated until all training patterns are processed and a stable family of clusters is generated.

The central part of the network computes the matching score (Euclidean distance) reflecting the degree of similarity between the present input and the previously encoded clusters. The MINNET is used to identify the cluster most similar (the minimal Euclidean distance) to the present input vector. The MINNET calculates the Euclidean distance between the present input vector and all existing clusters, and then identifies the cluster with the minimal distance. The similarity between this cluster and input pattern is then measured.

Since the proposed neural-net algorithm contains no hidden layers (flat net), the network structure depends on the type of the input data set. The number of neurons in the input layer is determined by the length of the input vector.

This network configuration can easily be modified to include voltage signals in addition to current signals.

Figure 4 graphically presents the outcome of the neural-net training. It shows a family of homogeneous

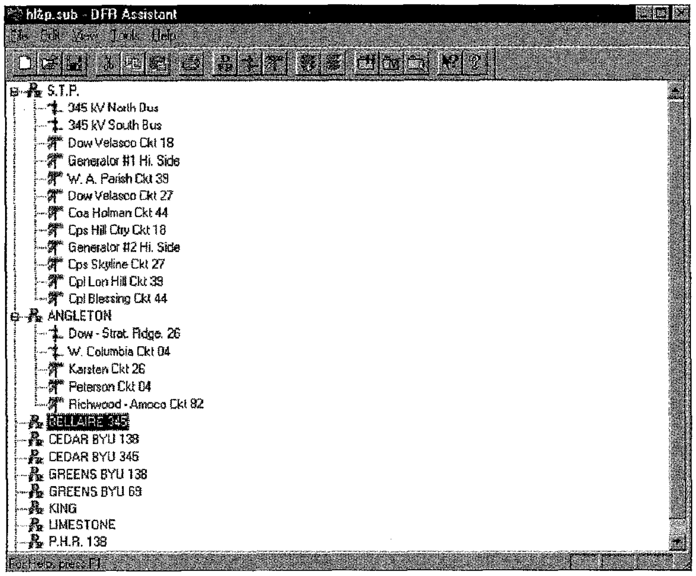


Figure 6. DFR Assistant configuration

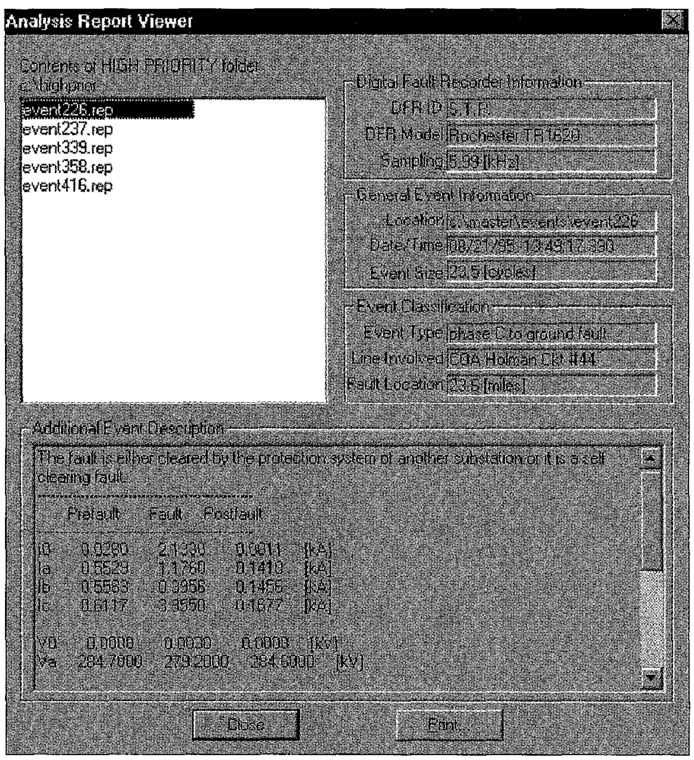


Figure 7. Example of analysis report

clusters, each labeled according to the type of fault pattern it contains.

Once the network is trained, it can be used as a general and very fast fault classifier. Every cluster generated in the clustering process is assigned a label according to type of fault pattern of its cluster members. A new fault pattern that needs to be classified is processed according to the following logic.

- Euclidean distances between a pattern and all of the clusters are calculated.
- The cluster with the minimal Euclidean distance is identified (if N nearest neighbors classification scheme is used, then N clusters with minimal Euclidean distance are selected).
- A new pattern is assigned class membership according to the closest cluster (or voting system is introduced in the case of N nearest neighbors).

Preliminary tests of the neural-net fault classifier were conducted using two different power system segment models. The neural-net performance was excellent, and average correct classification rates were above 90 percent. The results shown in Table 1 were obtained after neural net was trained using 1,200 different fault patterns.

During the testing, the neural net was presented with 295 new fault patterns. The neural network never *saw* these patterns, and its task was to classify new patterns based solely on the previous experience (i.e., using the information *learned* during the training).

Classification was based on the following logic:

- If a new pattern *belongs* to the nearest cluster (i.e., Euclidean distance between the pattern and the cluster center was smaller than the radius of that cluster), then the pattern inherited class membership of that cluster.
- If a new pattern *does not belong* to the nearest cluster (i.e., Euclidean distance between the pattern and the cluster center was greater than the radius of that cluster), then the pattern was assigned a label according to the class membership of the nearest three clusters (three nearest neighbors).

The results are summarized in Table 1.

Table 1. Neural network classification results

Correct Classification			Incorrect Classification			Correct (%)
Inside Cluster	Outside Cluster	Total	Inside Cluster	Outside Cluster	Total	
218	65	283	7	5	12	95.93

Application in Automated Fault Analysis

The neural network is designed to be incorporated in a PC-based system for automated transmission line fault analysis, called DFR Assistant. This system analyzes data files coming from the digital fault recorders (DFRs) located in substations. DFRs are monitoring selected currents, voltages, and contacts (e.g., relay trip status, circuit breaker status, etc.) and taking a snapshot of these signals when some abnormal conditions (faults) happen. This snapshot is recorded in a binary file that is transferred to a central location so that protection engineers can analyze it. Since any given utility company can have numerous DFRs, and, due to the "ripple effect", depending on the severity of a system disturbance, large numbers of DFR records may be generated. This places a burden on the protection engineers to manually examine all DFR records, identify records that are most important for a given case, and then analyze them. The new software facilitates the analysis of DFR recordings by automatically classifying all records into groups based on selected criteria. The records are filtered according to following criteria:

- Fault detected and identified, protection system operated within clearing time
- Fault detected and identified, protection system operation exceeds allowed clearing time
- Fault detected and identified, protection system did not operate at all
- No fault detected.

Based on these filters, records are archived in different *folders* (e.g., high priority folder, medium priority folder, and low priority folder) for further analysis by the protection engineers. Once configured at the central location, the software will monitor all DFRs and detect the arrival of the new recordings. Once the complete recording is being transferred, the analysis is started.

Figure 5 shows a hybrid system consisting of a supervised clustering neural net and a rule-based expert system. The neural net operates only on analog quantities of a DFR event file. It detects the faulted line and assigns the appropriate class membership to the fault (e.g., phase A to ground fault). The rule-based expert system uses contacts data to extract the timings of the operating sequence of the relays and breakers, and then processes these timings to verify the correctness of the overall operation of the protection equipment for a given event. At the end, a textual report is generated. The advantage of using neural networks is a reduction of the rule base (no rules are needed for dealing with the analog quantities), hence reduction in the number of thresholds that have to be prespecified. Also, an increase in processing speed is significant, since the neural network processing is parallel.

The role of the neural net in this solution is to process current signals and identify the one with the largest disturbance. Furthermore, it will try to classify disturbance according to the fault type.

Figure 6 shows the basic objects (i.e., DFRs and monitored bus voltages and transmission line currents) that are configured. The user must define the number and type of analog and digital channels that are monitored by a given DFR system. Usually, the practice of U.S. power utility companies is to record bus voltages (all three phases plus zero sequence) and at least two phase currents for each transmission line. In addition, relay trip signals, breaker contacts, and some communication channels are usually recorded. Once the configuration is finished, the incoming data files can be analyzed. The analysis is done automatically, and results are stored in different folders, depending on the type of event.

A neural network is a parallel, distributed, information processing structure consisting of processing elements interconnected together with unidirectional signal channels called connections

Figure 7 shows an example of an analysis report that was generated automatically by the program. The list box in the upper left corner lists all events that are processed and stored in the high-priority folder. The *digital fault recorder information* group box gives basic system parameters about the recorder that made the recording. The *general event information* group box contains data about a given event (its location in the archive, date/time stamp, and size in cycles). The *event classification* group box shows fault type, the transmission line involved, and the fault location. Finally, in the *additional event description* box, the user can see rms values for currents and voltages on the faulted transmission line during the time of the disturbance. This box also contains information on the operation of the protection relays and circuit breakers, if applicable to a given fault.

For Further Reading

A.G. Phadke, J.S. Thorp, *Computer Relaying for Power Systems*, John Wiley & Sons, Inc., New York, 1988.

M. Kezunovic, I. Rikalo, C.W. Fromen, D.R. Sevcik, "Expert System Reasoning Streamlines Disturbance Analysis," *IEEE Computer Applications in Power*, Volume 7, Number 2, April 1994.

M. Kezunovic, I. Rikalo, D.J. Sobajic, "Real-Time and Off-Line Transmission Line Fault Detection and Classification with Neural Nets," *International Journal of Engineering Intelligent Systems*, Volume 4, Number 1, March 1996.

M. Kezunovic, I. Rikalo, D.J. Sobajic, "High Speed Fault Detection and Classification with Neural Nets," *Electric Power Systems Research Journal*, Volume 35, Number 1, 1995.

Biographies

Mladen Kezunovic received his Dipl. Ing. degree in electrical engineering in 1974 and MS and PhD degrees from the University of Kansas, in electrical engineering in 1977 and 1980, respectively. His industrial experience is with Westinghouse Electric Corporation in the United States, and the Energoinvest Company in Sarajevo. His academic experience is with the University of Sarajevo and Washington State University. He has been with Texas A&M University since 1987 and is a professor. He is senior member of the IEEE, member of the IEEE PSRC, member of CIGRE, and a registered professional engineer in Texas. He is chair of the PES PSRC working group F-8 on Digital Simulator Performance Requirements and working group D-10 on Application of Intelligent Systems in Protection Engineering.

Igor Rikalo received his Dipl. Ing. degree from the University of Sarajevo and MS degree from Texas A&M University, all in electrical engineering in 1992 and 1994, respectively. Currently, he is working for TLI, Inc. as a systems engineer and for Texas A&M University as a research engineer. His main interests are in the area of intelligent system applications to power systems.