Finite Non-Deterministic Automata for Fault Diagnostic in Power Systems

RICARDO LINDEN[1]
VICTOR NAVARRO ARAÚJO LEMOS DA SILVA[2]

[1,2]CEPEL
Av. Horacio de Macedo
P.O. Box 354 - Cidade Universitária
21941-911 - Rio de Janeiro (RJ) - Brazil
(rlinden,navarro)@cepel.br

[1]Faculdade Salesiana Maria Auxiliadora
Rua Monte Elísio, s/n - Visconde de Araújo
27943-180 - Macaé (RJ) - Brazil

Abstract. This paper introduces an application based on finite non-deterministic automata for power systems diagnosis. Automata for the simpler faults are presented and the proposed system is compared with an established expert system.

Keywords: Finite Automata, Power Systems, Fault Diagnosis.

(Received June 18th, 2012 / Accepted July 9th, 2012 - *invited paper*)

1 Introduction

The current operational model for power systems demands the development of newer more sophisticated techniques for supervision, control, protection and automation in order to guarantee a reliable and safe operation.

When disturbances in power systems occur, the amount of alarms is high, making it difficult for the operator to determine the causes of those disturbances (make a diagnostic) and determine the correct actions that will normalize the systems. Due to these difficulties, there is a growing interest in the development of computational systems that may help the operator in making the diagnostic of the underlying cause of all the alarms.

Fault diagnostic is the name of the process of discovery of the cause of disturbances, that is, of the behavior of power systems that are not those expected in normal situations. This is a field of study that has deserved large attention and there are several articles that apply different techniques to solve them, such as fuzzy logic, either standing alone [3] or together with expert systems [8], neural networks [13] and [10], genetic algorithms [4], decision diagrams [7], Bayesian models [9] and expert systems [12] and [1]. If we extend our research to similar areas, we can find new applications of agent based systems to process fault diagnosis [11].

In this paper we propose the application of finite automata for fault diagnostic in power systems. Since we need empty transitions to describe alarm loss, the automata we are going to use are the finite non-deterministic ones. This is a natural extension of the work described in [5], where we first considered this idea.

This paper is organized as follows. In section 2, we describe finite automata, in section 3 we apply them to a typical subset of power systems and in section 4 we conclude with some final remarks and future directions.

2 Finite Automata and State Transducers

Finite automata are a useful model for many hardware and software elements. They serve as system models that can be at any moment at a finite number of states, elements that define the current characteristics, besides memorizing part of the history of the system [2]. As the system receives inputs, it makes the transition from one state to the other, until it stops receiving inputs. At this moment, if the system is at a state belonging to the set of final states, it considers the input set accepted.

Formally, a finite automata is described by five characteristics:

1. a finite set of states, *K*,
2. a finite input alphabet, Σ,
3. a set of transitions, δ,
4. an initial state, S, where, $S \in K$,
5. a finite set of final states, F , where $F \subseteq K$.

Hence, we can completely describe a finite automaton through a quintuple $M = \{K, \Sigma, \delta, S, F\}$.

Transitions are described by triples $\{s_i, \Sigma_T, s_f\}$ where s_i is the starting state of the transition, Σ_T is the set of symbols in the alphabet (characters) that trigger the transition when the current state is s_i and s_f is the new current state of the automaton after the transition. If for every element of Σ there is a single transition and no transition can be made over the empty input, that is, $\varepsilon \notin \Sigma_T$, then the automaton is called deterministic (FDA). In all other cases, it is called non-deterministic (FNDA).

Automata are usually represented as a directed graph, where states are represented as circles and final states by double edged circles and transitions are edges labeled with the symbols that trigger the transition between the two connected states (Figure 1).

In this paper we will use a non deterministic finite state transducer (NDFST), that is very similar to a FNDA, but with the difference that its function is not to accept strings or languages, but rather transform input strings into output ones. In each state the NDFST outputs (or writes into an output tape) a string of symbols, as a function of the current state and of the current input symbol [2]. In the case of the NDFST described in this paper, all non final states output a string of zero symbols, while the final states output the diagnostic of the fault they model. In all figures that describe the NDFST created, output strings will be omitted for the sake of simplicity and ease of understanding.

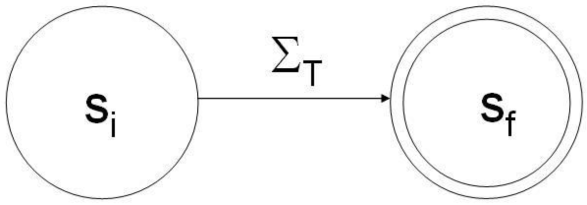

Figure 1: Graphical representation of the transition $\{s_i, \Sigma_T, s_f\}$. The circle of state sf has double edges because it belongs to the set F of final states.

3 Application to Power Systems

Power systems have a hierarchic structure. There are several substations that are connected through power lines and some may supply power to final customers. Each substation has several circuit breakers in its outlets that serve to isolate the power distribution lines, or feeders. All these feeders leave the substations and carry electric power to the final customers.

In this section we will describe the application of the NDFST to a typical section of a transmission line that is shown in Figure 2. This section represents a bar to which several feeders are attached through smaller circuit breakers that can isolate them and that have the ability of automatically turning on. Typically this bar is isolated from the rest of the system by a circuit breaker that can protect it.

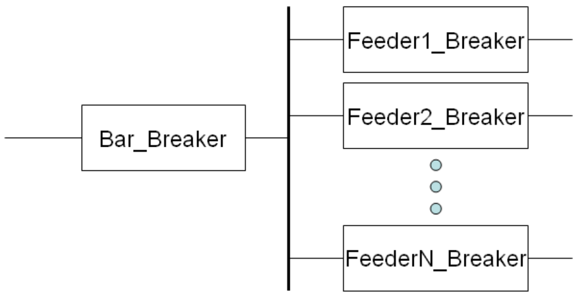

Figure 2: Typical section of a power transmission line. Circuit breaker *Bar_Breaker* protects the bar, to which are connected the feeders that are duly protected by smaller circuit breakers called *Feeder_Breaker*, *Feeder2_Breaker*, ..., *FeederN_Breaker*.

There is a fault in one of the feeding lines that takes to a sequence of alarms given in the following sequence:

1. triggering of the feeder protection: indicates that there is a fault at the power line;

2. definitive triggering: indicates that the fault is going to make the breaker open;

3. breaker opening: indicates that the breaker has opened.

There is a naive model that can render this situation, which consists simply of a starting state and one state for the indication that each one of those alarms has occurred. The state after the breaker opening would be final (for the diagnostic has been reached) and would output the message consisting of the diagnostic at hand.

Notice that the fact that the state is final does not mean that the operation has ceased. The truth is exactly the opposite: the system needs to operate continuously and any new alarms must be processed. Hence, we need to create an ε-transition from the final state to the initial one in order to model this continuous operation mode.

Another important characteristic that was added to this version is the ability to deal with possibly spurious alarms. If an alarm is not followed by the ones expected in this fault, it probably was generated spuriously, either by communication fault, contact error or any other reason. Therefore, we create a time based transition that will model the amount of time elapsed since we entered in a state. If the next expected alarm does not come, we will perform the transition to the initial state and consider that the alarm received is a spurious one.

The NDFST that follows this naive implementation is shown in Figure 3.

Unfortunately, this model is as simple as it is inaccurate. In real systems, alarms can come out of order, because their occurrence is very close in time and natural delays may cause them to come together or in reverse order. Worse still, they can even be lost, due to errors in communication or even errors in the protection system. Even if this situation has a small probability associated with it, it is important to be considered when building a diagnostic system.

Therefore, we need a NDFST that is capable of recognizing a fault when alarms come out of order and even when one of the alarms is lost. That is performed by including ε-transitions to our NDFST â their existence implies that either we are waiting for an alarm or we have already accepted the fact that it will not come at al.

We also need to accept the situation when alarms arrive out of order. This is achieved by creating alternative paths â that is, the system can either receive one of the alarms first or the other. Notice that we need different states for each path, for the states indicate the memory of the alarms received and the expectations we have towards incoming alarms.

In Figure 4 we can see one NDFST that shows this characteristic and is capable of rendering the correct diagnostic in case of the loss of either the protection trigger or the definitive trigger alarm.

The loss of both alarms was not considered due to the small probability of this occurrence, but to model it we only need to include a ε-transition from state q_0 to state q_3. The loss of the alarm that informs the opening of the circuit breaker is not considered; in this case, it is necessary the definition of the occurrence - without this definition, we are probably describing another fault, as we will see below.

There are bar faults that cause the same sequence of events in the feeders, so they can be described precisely by the automata that are similar to the ones described in Figure 4. Nevertheless, there is a second kind of fault, namely, bar isolation due to feeder problem that must also be considered.

In this case, a feeder fault triggers the protection and the definitive trigger, but due to any number of reasons such as mechanical faults, the circuit breaker does not open and does not isolate the faulty feeder. The fault then propagates to the bar, causing its circuit breaker to open.

The NDFST that is capable of recognizing this situation is similar to the one shown in Figure 4 with the change of the input that makes the transition from state q_3 to state q_4 is the opening of the bar breaker *(Bar_Breaker1_Open)*.

If we wish to model both faults simultaneously in a single transducer, we have the NDFST described in Figure 5. This is similar to the NDFST described in Figure 4 with the addition of two states: q_5, which represents the bar disconnection due to feeder fault and q_6, that represents the delayed opening of the feeder breaker. In this situation, we will have a situation that models the case where we have two simultaneous diagnostics: loss of feeder due to a feeder fault (q_6) and loss of bar due to feeder fault (q_5).

The delayed alarm probably indicates that we have two independent faults (bar loss and feed loss). This is due to the fact that if we simply lost the alarm, the bar would not open, for its breaker would not "sense" the fault already isolated by the feeder breaker. Hence, some knowledge on the power system must be applied in order to avoid generating diagnostics that are neither helpful nor correct.

This is an important feature because no knowledge system should be generated in an entirely automated fashion. Even if topological information is available and a basic automaton generated, the user should consider review any result achieved, in order to guarantee that the model reflects the real situation that the operator is facing.

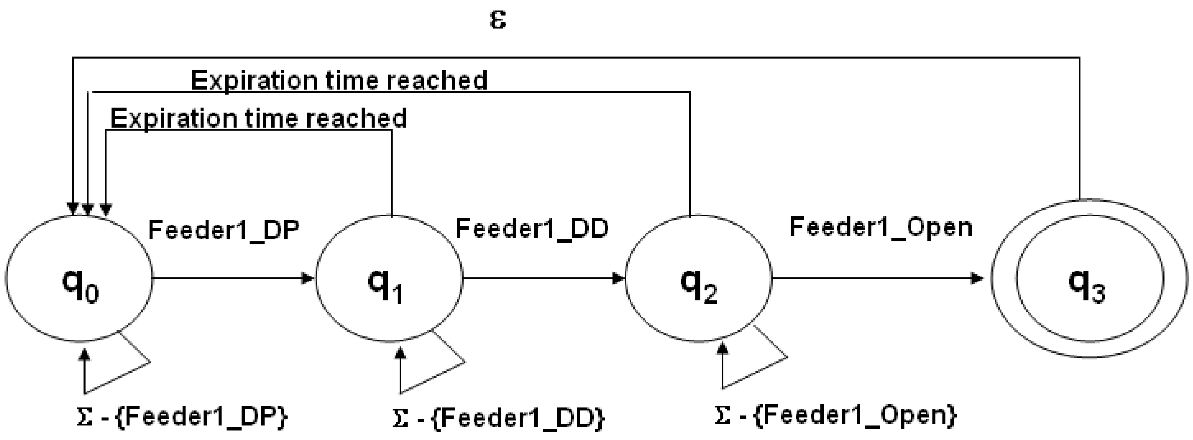

Figure 3: The NDFST that can model a fault in one of the feeder lines. *Alim1_DP* is the alarm that indicates triggering of protections at feeder1, while *Feeder1_DD* is the alarm that indicates that the definitive triggering has occurred and Feeder1 Open is the alarm that indicates that the breaker is open. The state q_3 outputs the diagnostic ("Fault at feeder 1"). Notice that after emitting the diagnostic the system is still able to analyze incoming alarms, which is modeled by the ε-transition leaving the final state and arriving at the initial one. We also included in this model the time based transitions from the intermediary states that will model the fact that those alarms may have come spuriously. Since the circuit breaker is only sensitive to alarms relating to its faults, all other alarms are ignored. This can be also modeled by the edges that do not cause a change of state and that are label with all alarms except those that interest the current state. These edges will be omitted in the next drawing for ease of understanding, but their existence is implicit.

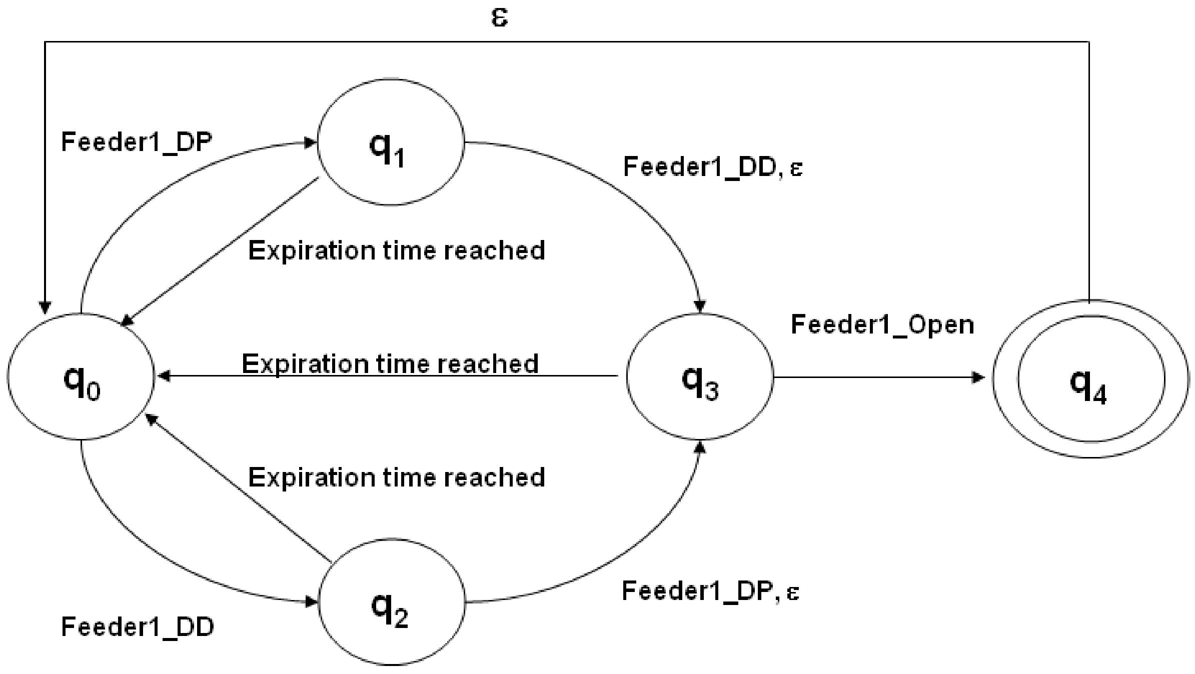

Figure 4: NDFST capable of modeling the fault in one of the feeder, taking into account a possible loss of either the protection trigger or definitive trigger alarm. Transitions to the same state labeled by all other alarms have been omitted for simplification.

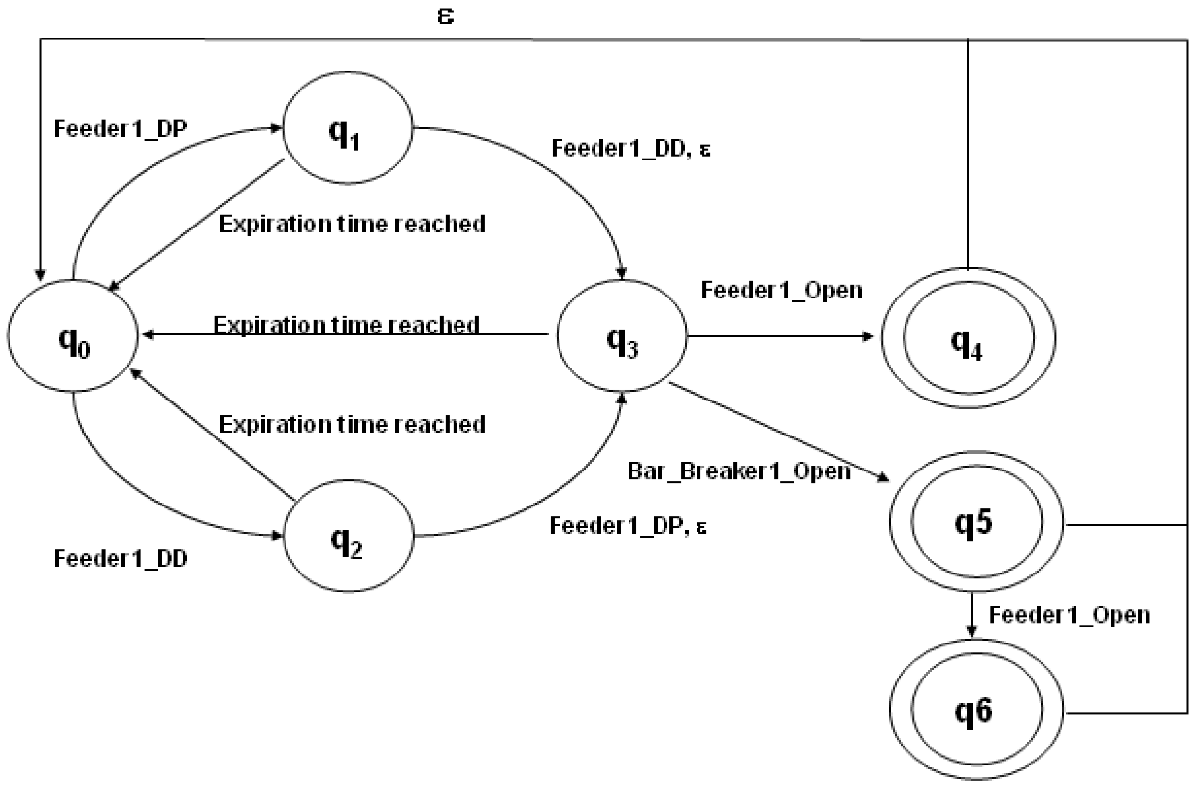

Figure 5: NDFST capable of modeling the fault in one of the feeder, taking into account a possible loss of either the protection trigger or definitive trigger alarm. Transitions to the same state labeled by all other alarms have been omitted for simplification.

In order to conclude the model, it is necessary to put the automaton of Figure 5 (which we will call L_1) together with the automaton of Figure 4, with the alarms associated with the bar circuit breaker (which we will call L_2), because they must be analyzed in parallel. This is done by creating the automaton $L_1 \cup L_2$, whose formal definition is given by $M = \{K, \Sigma, \delta, S, F\}$, that is defined by:

1. $K = K_{L_1} X K_{L_2}$, where X indicates the Cartesian product;

2. Σ = all possible alarms;

3. $S = (q0_{L_1}, q0_{L_2})$, an initial state that is the state formed by the both initial states;

4. F = all states in K that contain at least one final state in its pair;

5. δ = the transition function that is defined as follows: $\delta((q_{L_1}, q_{L_2}), a) = (p_{L_1}, p_{L_2}) \iff \delta(q_{L_1}, a) = p_{L_1} \wedge \delta(q_{L_2}, a) = p_{L_2}$, where it is possible that $q_{L_1} = p_{L_1} \vee q_{L_2} = p_{L_2}$.

The problem with this definition is that the number of states in the resulting automaton is equal to the product of the number of states in each of the previous automata, what implies in an extremely large number of states for a system that has a larger number of feeders. In our simple example, we would have 24 states to model only two faults.

There are some reductions that could be made manually to the automaton generated, but in order to allow for an automatic implementation (yet to be started), these reductions where not considered in our system. One example of these is the elimination of state q_6 in Figure 5, allowing the state q_5 to transition into state q_4 when the alarm *"Feeder1_Open"* occurred. This reduction is possible because both states are terminal and emit the same diagnostic, but does not represent a large gain either in performance or memory consumption that would create an advantage large enough to prevent the automatism of the model presented so far.

The advantage of a hierarchical system is that faults at feeder i do not affect the feeder $j, \forall i, j = 1, ..., n, i \neq j$. Hence, we can create a state transducer similar to the one described above for each of the feeders. Therefore, the total number of states in the system is not a product of the states in each feeder, but a sum of them, which is several degrees of magnitude smaller. Besides, the state transducer in each feeder can be evaluated in a time of an order linearly proportional to the number of existing alarms.

The feeder independence allows for a completely independent implementation, with a thread for each transducer, resulting in a possibly parallel implementation with a thread in a different machine or processor, without the needed of shared memory of communication among automata.

4 Improvements to our previous work

Spurious alarms are a natural thing in SCADA systems. They can be due to problems in sensors, problem in communication or simply by a transient problem in the electrical contacts that detect a fault situation. Hence, we can receive alarms that stand alone and do not really indicate a fault, but a normal situation that will resolve itself. If the full sequence that allows for a disgnostic does not occur, we can consider that the alarm was spurious and expired.

Our previous work [5] did not consider this situation. Hence, we included in this expanded model time based transitions, that account for an expiration of single alarms. This time based transition is important to model an important and common reality in power systems and allows for a more complete and realistic diagnostic system.

The new model also includes an ε-transition from the terminal diagnostic state to the initial one. This is important because it allows for continuous operation - the automaton should not cease its operation when a diagnostic is reached, for new faults may arrive in the future. Hence, after emitting the diagnostic, we need to return to the initial state in order to be able to analyze new incoming alarms. Our previous system did not have this capability and the implemented model would be "stuck" in the terminal state and not be able to emit new diagnostics after its first run.

5 Comparison to other diagnostic systems

NDFST have simple implementations with lightweight programs. The control logic can even be implemented by an udegraduate student as classwork. Hence, its value relies in the NDFST itself, not in the underlying mimplementation.

In the case of this work, we have several NDFST that describe faults in a transmission system. Each group of faults is independent and hence, the automata can be separated and work in parallel. Therefore, we can feed each automaton to a lightweight process implementing a NDFST and all of them can run in parallel, allowing for a massively parallel implementation of our disgnostic system.

The expert system described in [6] has a single threaded implementation. This is usual in this kind of system for they have an inference model whose working is computationally costly and due to its sequential nature it is impossible to make it completely parallel. It would be possible to make the knowledge base parallel, but given the inference engine is so "heavy", it might be hard to put several of them to work simultaneously.

Every transition in a NDFST is completely clear and understandable by a human. Therefore, in order to explain a dignostic, an operator needs only to verifiy the path followed by the NDFST before emitting a diagnostic. This is very useful especially in the first stages of operation, for the operators usually take some time before developing confidence in the automatic diagnostic tool. Besides, this capability is very useful for auditing and training, for post mortem discussions can be made over every fault ocurred.

This clarity is an advantage over black box system, such as neural networks. which can be considered as black box systems. Even though they are usually precise in their diagnostics, they lack the explanation ability that is inherent to both expert systems and NDFST. Besides, given that neural networks are mathematical relations, they must be trained, lacking the capacity to absorb previous knowledge, that is abundant in the area of fault diagnostic.

This kind of system is not naturally amenable to fuzzy logic treatment. In power systems diagnostic is made on alarms that are binary in nature (either they ocurred or not), and not based on continuous quantities that can be interpreted with linguistic terms.

6 Conclusion and future work

In this paper we proposed a model based on non deterministic finite state transducer for the problem of fault diagnostic in power systems. Due to the nature of the transducers, time between alarms and diagnostics is short, for it is a linear function of the number of received alarms.

This system marks an improvement from our initial work in [5] by the inclusion of time based transitions and by the ε-transition that models the need for continuous operation. One important characteristic of those transitions is that they are always present at fixed points in the automaton. Hence, they can be generated automatically, making it easier for the user to create models in any upcoming design tool.

It is important to point out that each automaton can work independently from other automata in the system, allowing for a massively parallel implementation and giving it a capacity for optimal performance, even when the power system is extremely complex, which is an advantage over single threaded expert systems, as described in [6].

The system also shows an advantage over neural networks and other black box systems because it allows for previous knowledge to be inserted in its model and does not rely solely on automatic training. This is a feature that becomes important if convoluted situations may require extraordinary diagnostics.

As future work, we intend to start implementing an editor and the daemon program that will incorporate the ideas described in this paper into a full fledged diagnostic tool integrated to a SCADA system.

References

[1] ANGELI, C. Online expert systems for fault diagnosis in technical processes. *Expert Systems*, 25(2):115–132, May 2008.

[2] HOPCROFT, J. E., MOTWANI, R., and ULLMAN, J. D. *Introduction to Automata Theory, Languages and Computation*. Addison Wesley Pub., Londres, 3 edition, 2006.

[3] HUANG, Y. C., YANG, H. T., and HUANG, C. L. Developing a new transformer fault diagnosis system through evolutionary fuzzy logic. *IEEE Trans. Power Delivery*, 12:761–767, 1997.

[4] LI, Y., BAO, D., LUO, C., Huang, L., and Cao, L. An intelligent fault diagnosis method for oil-immersed power transformer based on adaptive genetic algorithm. *Lecture Notes in Electrical Engineering (Part I) (Advances in Automation and Robotics, v.1)*, 122:155–162, 2012.

[5] LINDEN, R. and SILVA, V. N. A. L. Autômatos finitos não-determinísticos para diagnóstico de falhas em sistemas de potência. *Revista de Sistemas de Informação da FSMA*, 1:1, 2009.

[6] LINDEN, R., SILVA, V. N. A. L., and RIBEIRO, G. F. A framework for expert systems development integrated to a SCADA/EMS environment. In *Proceedings of the 14th International Conference on Intelligent Systems Application to Power Systems*, December 2007.

[7] MOORE, M., MONEMI, S., and WANG, J. Integrated diagnostics for electric utilities. In *World Multiconference on Systemics, Cybernetics and Informatics*, pages 458–463, Orlando, July 2000.

[8] NAN, C., KHAN, F., and IQBAL, T. Real-time fault diagnosis using knowledge-based expert system. *Process Safety and Environmental Protection*, 86(1):55–71, January 2008.

[9] NIE, Q. and WANG, Y. An augmented naive bayesian power network fault diagnosis method based on data mining. In *Proceedings of the 2011 Asia-Pacific Power and Energy Engineering Conference*, 2011.

[10] NIU, Q., WANG, L., and SHI, Y. Realization of a power transformer on-line monitoring and diagnosis system based on dga and pnn. In *International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE)*, 2010.

[11] PERK, S., TEYMOUR, F., and CINART, A. Adaptive agent-based system for process fault diagnosis. *Ind. Eng. Chem. Res.*, 50(15):9138–9155, June 2011.

[12] SILVA, V. N. A. L., RIBEIRO, G. F., LINDEN, R., and et al. Sistemas inteligentes híbridos para diagnose no coi da cosern. In *Anais do VII Seminário Técnico de Proteção e Controle*, June 2003.

[13] WANG, Z., LIU, Y., and GRIFFIN, P. J. Neural net and expert system diagnose transformer faults. *IEEE Computer Applications in Power*, 13(1):50–55, January 2000.