Proceedings of the V Brazilian Conference on Neural Networks - V Congresso Brasileiro de Redes Neurais
pp. 31-36, April 2-5, 2001 - Rio de Janeiro - RJ - Brazil

Neural Networks to Transient Stability Analysis of Electrical Power Systems

Carlos R. Minussi, Lilian Milena Ramos, Sandra Cristina Marchiori,
Mara Lúcia Martins Lopes, Anna Diva P. Lotufo
E-mails: minussi@dee.feis.unesp.br, lilian@dee.feis.unesp.br, sandra@dee.feis.unesp.br,
mara@dee.feis.unesp.br, annadiva@dee.feis.unesp.br

Abstract

The objective of this work is the development of a methodology for transient stability analysis of electric power systems (critical time determination for short circuit faults) based on a neural network. Here, it is used back-propagation algorithm with an adaptive process based on fuzzy logic. This methodology results in a fast training, when compared to the conventional formulation of back-propagation algorithm. The input is composed by the nodal active and reactive electrical power vectors and data about the fault, represented in a similar way to the binary code. The output (critical time) is determined by a computational program based on a hybrid methodology – simulation / Liapunov Direct Method. It is a less dimension network when compared to those ones found in the literature, however it needs a low computational cost in the execution training. To illustrate the proposed methodology, an example is presented which considers a multi-machine system composed of 10 synchronous machines, 45 buses, and 73 transmission lines, based on the configuration of a southern Brazilian system.

1. Introduction

The Electric Power Systems transient stability analysis can be done, for example by simulation, i.e., by numeric solution of differential equations that describe the system dynamic, and after, analyzing the synchronous machine oscillation curves.

Considering the great number of faults to be analyzed, either the simulation or the Liapunov Direct Method are alternatives that offer no total conditions yet to real time applications. Then, this work proposes an investigation to use the Artificial Neural Networks in the transient stability diagnosis.

To obtain the desired results, i.e., the network presents ability to do complex diagnosis as in electric power system problems, the so called network as a configuration formed by units called neurons displayed in layers and forming complex interconnections ([5, 10]). These interconnections are formed by weights that have to be adjusted in function of a set of patterns to produce desired outputs. This activity is defined as being the training or learning and it is done off-line. Once, it is adjusted, the network will be able to do diagnosis with satisfactory precision considering patterns that do not belong to the training set (generalization ability). This diagnosis can be done with no computational costs, which can be a possibility to a real time analysis.

In this work the neural network training will be done by Back-propagation technique (BP) ([10]). The BP algorithm is considered a benchmark in precision. However, its convergence is quite slowly. Then, the proposal of this work is to adjust the training rate g during the convergence process, to reduce its execution time. The g adjustment is done using a fuzzy controller. We also use a decaying exponential function that establishes a priority in the regulator actuation in the initial training time and avoid instability in the convergence process. The system dynamic model adopted in this work is the classical model ([1, 7]), objecting, only, a reduction in the computational costs to generate the training data. However, it can be used more elaborated models without changing the proposed architecture. The input is compounded by the nodal active and reactive electrical power vector and the fault data. The fault data is represented in a similar way to the binary code. The output (critical clearing time) is determined through a computational program based on a hybrid methodology using simulation / Lyapunov Direct Method ([4]). That is a network of reduced dimension when compared to the main proposals found in the literature. For this reason it has a smaller computational cost to perform the training.

To illustrate the proposed methodology, an example is presented which considers a multi-machine system composed of 10 synchronous machines, 45 buses, and 73 transmission lines, based on the configuration of a southern Brazilian system.

2. System Model

Considering an Electrical Power System composed by *ns* synchronous machines, the dynamical behavior of the *i*-th machine, related to Center of Angles (CA), is described by the following differential equation (classical model) ([1, 7]):

$$M_i \, \ddot{q}_i - P_i(q) = 0, \; i \; \hat{I} \; N \qquad (1)$$

where:

$$P_i(q) = Pm_i - Pe_i - (M_i \, PCOA) / MT; \qquad (2)$$

$M_i = 2 \, H_i / ws;$

$ws \triangleq$ synchronous speed of the rotor

	=	$2pf_0$;
H_i	=	inertia constant (s);
f_0	=	nominal frequency of system (Hz);
q_i	$\underline{\Delta}$	rotor angle of i-th synchronous machine related to CA (in elect. radians)
	=	$d_i - d_0$;
d_i	=	rotor angle of i-th synchronous machine in relation to synchronously rotating reference frame (in elect. radians);
d_0	=	$\sum_{j \in N} M_j\, d_j$;
Pm_i	=	mechanical power of input (pu);
Pe_i	=	electrical power of output (pu);
$PCOA$	$\underline{\Delta}$	accelerating power of CA
	=	$\sum_{j \in N} (Pm_j - Pe_j)$;
MT	=	$\sum_{j \in N} M_j$;
N	$\underline{\Delta}$	index set of synchronous machines that comprise the system
	=	$\{1,2, \ldots, ns\}$.

3. Neural Network Structure

The i-th output element (neuron) ([10]) is a linear combination of the element inputs x_j that are connected to the element i by the weight w_{ij}:

$$J_i = \sum_j w_{ij}\, x_j \tag{3}$$

Each element can have a bias w_0 fed by an extra constant input $x_0 = +1$. The linear output J_i is finally converted in a non linearity as a sigmoid ([5, 10]) and relay ([5, 10]), etc. The relay functions are appropriated for binary systems, while the sigmoid functions can be employed for both continuous and binary systems.

4. Neural Network Training

The BP training is initialized by presenting a pattern $\boldsymbol{X}$ to the network that will give an output $\boldsymbol{Y}$. Following it is calculated an error in each output (the difference with the desired value and the output). The next step is to determine the back propagated error by the network associated to the partial derivative of the quadratic error of each element related to the weights, and finally adjusting the weights of each element. Then a new pattern is presented, and the process must be repeated until convergence ($|$error$|$ $\leq$ arbitrated tolerance). The initial weights are usually adopted as random numbers ([10]). The BP algorithm consists in adapting the weights such that the network quadratic error is minimized. The sum of the instantaneous quadratic error of each neuron of the last layer (network output) is given by ([10]):

$$e^2 = \sum_{i=1}^{no} e_i^{\,2} \tag{4}$$

where:

e_i	=	$d_i - y_i$;
d_i	=	desired output of the ith element of the last layer ;
y_i	=	output of the i-th element of the last layer;
no	=	number of neurons of the last layer.

Considering the i-th network neuron and using the descent gradient ([5, 10]) method, the weight adjustments can be formulated by ([10]):

$$\boldsymbol{V}_i(r+1) = \boldsymbol{V}_i(r) + \boldsymbol{F}_i(r) \tag{5}$$

being:

$F_i(r)$	=	$-\,g\,[\tilde{N}_i(r)]$;
g	=	stability control parameter or training rate;
$\tilde{N}_i(r)$	=	quadratic error gradient related to neuron i weights;
$\boldsymbol{V}_i$	$\underline{\Delta}$	vector containing neuron i weights
	=	$[\,w_{0i}\;\; w_{1i}\;\; w_{2i}\; \ldots\; w_{ni}\,]^T$.

The adopted direction in equation (5) to minimize the objective function of the quadratic error corresponds to the opposite gradient direction. The γ parameter determines the vector length F_i. Considering that this work deals with transient stability analysis (determination of critical clearing time) (the values are always positive), the nonlinear function to be used is the sigmoid function defined by ([5, 10]) (varying between 0 and +1):

$$y_i = 1 \,/\, \{(1 + exp\,(-\,l\; J_i)\} \tag{6}$$

being:

l	=	constant that determines the function y_i slope.

Then, calculating the gradient as shown in equation (5), considering the sigmoid function defined by equation (6) and the momentum term, it is obtained the following weight scheme ([5, 10]):

$$v_{ij}(r+1) = v_{ij}(r) + Dv_{ij}(r) \tag{7}$$

where:

$Dv_{ij}(r)$	=	$2\,g\,(1-h)\,b_j x_i + h\,Dv_{ij}(r-1)$;	(8)
v_{ij}	=	weight corresponding to the connection with the i-th and the j-th neuron;	
g	=	training rate;	
h	=	momentum constant ($0 \pounds h < 1$) ([10]).	

If the j-th element is in the last layer, then:

$$b_j = s_j\, e_j \tag{9}$$

being:

s_j	$\underline{\Delta}$	sigmoid function derivative, given by equation (6), related to J_j	
	=	$l\, y_j\,(1 - y_j)$.	(10)

If the j-th element is in other layers, we have:

$$b_j = s_j \sum_{k \in S(j)} w_{jk}\, b_k \tag{11}$$

where:

$S(j)$	=	index set of the element that are in the next layer to the j-th element layer and are interconnected to the j-th element.

The g parameter used as a stability control of the iterative process is dependent of l ([6]). The network weights are randomly initialized considering the interval {0,1}.

By convenience, the parameter g (training rate) can be redefined by following:

$$g = g^* / l \qquad (12)$$

Replacing equation (12) in equation (8), it will be "cancelled" the amplitude dependency of s related to l. The s amplitude will be maintained constant to every l. This alternative is important considering that l will only actuate in the left and right tails of s. Then, equation (8) can be written as following:

$$Dv_{ij}(r) = \{ 2\, g^* (1 - h)\, b_j / l \}\, x_i + h\, Dv_{ij}(r-1). \qquad (13)$$

The BP algorithm is considered in the technical literature a benchmark in precision, although its convergence is very slow. In this way, this work proposes to adjust the training rate g^* during the convergence process objecting the reduction in the execution training time. The g^* adjustment is done by a proceeding based on a fuzzy controller.

The basic idea of the methodology consists in determining the system state, defined as the global error eg and the global error variation Deg, objecting a control structure that leads the error to zero in a reduced iteration number, when compared to the conventional procedures. In this work, the control is formulated using the fuzzy logic concepts. The global error εg and its variation Deg are the system state components, and Dg^* is the control action that must be done in the system. Initially, the global error is defined as:

$$eg = \sum_{j=1}^{np} \sum_{i=1}^{no} e_i^{\,2} \qquad (14)$$

where:

eg	=	global error of the neural network;
np	=	number of the network pattern vectors.

The global error corresponds to calculate all output errors (output neurons) considering all network pattern vectors. The training must be executed using procedure 2 (one iteration per pattern). The global error is calculated in each iteration and parameter g^*, adjusted by an increase Dg^* determined by fuzzy logic. The system state and the control action are defined as:

$$E^q = [eg^q \;\; Deg^q\,]^T \qquad (15)$$

$$u^q = Dg^{*\,q} \qquad (16)$$

being:

q	=	current iteration index.

For a very big input pattern $\boldsymbol{X}$, eg and Deg can saturate. Then, the adaptive control is done using an exponential decreasing function applied to the fuzzy controller response. In this way, the adaptive controller is done by:

$$Dg^{*\,q} = exp\,(-\,a\,q\,)\,Dy^q \qquad (17)$$

where:

a	=	an arbitrary positive number;
Dy^q	=	increase proceeding from the fuzzy controller in instant q.

This parameter will be used to adjust the network weight set referred to the subsequent iteration. The process must be repeated until training be concluded. It is a very simple procedure whose control system requests an additional effort, although reduced, considering that the controller has two input variables and only one output. This is an improvement of that one presented in reference [2], i.e., using the same variables eg and Deg to do the control. However, in this work there are introduced the following contributions: 1) improvement of BP algorithm proposal; 2) the proposal of the fuzzy controller is original (a set of rules and the use of an exponential decreasing function applied to the controller response).

Definition. Consider a set of objects B. Then, a fuzzy set A in B is defined as being a set of coordinated pairs ([5, 9]):

$$A = \{(b, m_A(b)) \mid b\ \hat{I}\ B\} \qquad (18)$$

being:

$m_A(b)$	=	membership function of b in A.

The membership function $m_A(b)$ shows the degree of how b is in A. The fuzzy control is basically formed by three parts: (1) *fuzzification* that converts real variables in linguistic variables; (2) *inference* that consists in the manipulation of rules using *if-then* statements, and fuzzy operations (AND, OR, NOT, etc.) and (3) *defuzzification* that converts the obtained results (linguistic variables) in real variables, which form the control action. The fuzzy membership functions can have three different forms, as: triangular, trapezoidal and gaussian, according to the preference/experience of the designer.

The most common defuzzification method is the centroid ([5, 9]) that finds the inertial center of the fuzzy set solution. We have the following equation considering a discrete fuzzy set ([5, 9]):

$$u = \Big(\sum_{i=1}^{nr} mi\, ai \Big) / \Big(\sum_{i=1}^{nr} mi \Big) \qquad (19)$$

where:

mi	=	value of the membership function;
ai	=	value of the set that contains a membership value;
nr	=	number of fuzzy rules.

The u value calculated by equation (19) corresponds to the inertia center projection of the figure defined by the rule set over the control variable axe. Each state variable must be represented between 3 and 7 fuzzy sets. The control variable must also be represented by the same number of fuzzy sets. The eg variable must be normalized considering as a schedule factor the first global error generated by the network, i.e., $q = 0$. With this representation, the variation interval will be between 0 and +1. If the adaptation heuristic is accordingly coincident the process convergence will be an exponential decreasing. The $\Delta\varepsilon g$ variable will vary between –1 and +1. If the convergence process is exponential decreasing, the $\Delta\varepsilon g$ values will always be negative. In this case, although the $\Delta\varepsilon g$ schedule is between –1 and +1, it must be employed in the rule set, an accurate adjustment between –1 and 0. In the other interval (0, +1], the adjustment can be more relaxed.

In the fuzzy controller the rules are codified as a decision table form. Each input represents the fuzzy variable value $\Delta\psi$ given the global error values εg and the global error variation $\Delta\varepsilon g$. The parameter g^* must be arbitrated in function of l (sigmoid function slope).

The variations γ^* also follow the same procedure.

In the fuzzy controller the rules are codified as a decision table form. Each input represents the fuzzy variable value $\Delta\psi$ given the global error values εg and the global error variation $\Delta\varepsilon g$. The parameter g^* must be arbitrated in function of l (sigmoid function slope). The variations g^* also follow the same procedure.

Table 1 shows the fuzzy rule set in a total of 30 rules. The number of rules can be increased to improve the network performance during the training.

Table 1. Fuzzy controller rules.

$\Delta\varepsilon g$	εg					
	ZE	PVS	PS	ME	PL	PVL
NL	ZE	ZE	NS	PL		PL
NS	NL	ZE	ME	NL	ME	
ZE	PS	NL	NL		PS	NS
PS	PL	ME		PL	NS	ZE
ME	ZE		PL	PS	ZE	NL
PL		ZE	NL	ME	ME	ZE

where:

NL	=	Negative Large;
NS	=	Negative Small;
ZE	=	Near to Zero;
PVS	=	Positive Very Small;
PS	=	Positive Small;
ME	=	Medium;
PL	=	Positive Large;
PVL	=	Positive Very Large.

To analyze the developed methodology performance, there are defined gains, considering the number of cycles and the necessary time to training time, in the following way respectively:

$$GC = NBP \,/\, NFBP \tag{20}$$

$$GT = TBP \,/\, TFBP \tag{21}$$

being:

NBP	=	BP number of cycles;
NFBP	=	fuzzy number of cycles;
TBP	=	execution time (processing) by BP (s);
TFBP	=	execution time by fuzzy BP (s).

5. Proposed Solution

The structure of the proposed neural network intends to study the Electrical Power Systems transient stability that corresponds to the determination of critical clearing time for contingencies like short-circuit. It is a non-recurrent network, basically compound by three layers (input, intermediary, and output) and neurons type sigmoid with soft slope (small l) – utilized as a way to reduce the training computational time. The input / output stimulus are defined considering the input made up for the nodal active and reactive electrical power vector and data of the fault, represented in a similar way to the binary code, and the output correspondent to the critical times, for a list of contingencies pre-defined, provided by a hybrid computational program ([4]) (simulation / Liapunov Direct Method).

5.1. Sigmoid Function

It is considered a sigmoid function defined by the equation (6). In the BP algorithm, the adaptation of the weights that connect each neuron is carried out using, basically, the error propagated in the inverse direction, multiplied by $s(J)$ (partial derivative of y with respect to J) and by the input in the referred neuron. In this way, the weights have effective adjustment only for values of J situated in the *central body* of $s(J)$ function. As the weights become significant, occurs a deceleration of its adjustments, having a tendency to a complete paralysis, exactly where meet together the ends of both *right* and *left tails* of $s(J)$ ([6]). In view of this, in the specific case of the critical time determination problem , it is proposed the use of sigmoid function relatively small l (inferior to 1), providing longer tails. This will permit a less restrictive choice of the network weights, if compared to the adopted in the bibliography. In a way this reduces the possibility of a paralysis occurrence and increases the speed of the BP algorithm convergence.

5.2. Input-Output Stimulus

It is proposed a neural network which pattern vector is defined by:

$$\boldsymbol{X} = [\boldsymbol{P}^T \boldsymbol{Q}^T L^T]^T \tag{19}$$

where:

$\boldsymbol{X}$	=	pattern vector;
$\boldsymbol{P}$	=	$[P_1 \; P_2 \; ... \; P_r]^T$;
$\boldsymbol{Q}$	=	$[Q_1 \; Q_2 \; ... \; Q_r]^T$;
P_i	=	active power of ith system bus ;
Q_i	=	reactive power of ith system bus ;
L	=	vector that contains the number of contingencies represented in a code like the binary code (–1,+1).

The choice of this binary representation is preferable in relation to (0,+1) representation, considering that the network input component "0" does not modify the weights. In this way (–1,+1) representation gives a faster convergence and consequently being more efficient. Each component of the L vector correspond to 1 bit of the representation in binary code. The pattern vector is a n-dimensional, and:

$$n = 2\,r + nb \tag{20}$$

where:

r	=	number of system buses ;
nb	=	number of bits that corresponds to the contingency number.

In this case, considering the classical model, only the active and reactive power vectors of the system bus is necessary for the evaluation of Electrical Power Systems transient stability – determination of critical clearing times. Considering the topology, the nodal voltages and the other parameters of the model (inertial constant, transient reactance, etc.) are kept unchanged, although the voltage can admit small variations in function of the change of the generation outline / system load. As the number of bits nb is considerately smaller than $2r$, then the proposed network is significantly smaller, if compared to those presented in the literature to solve transient stability problem: using back-propagation neural networks ([8]), and using Kohonen neural networks ([3]).

6. Application

There are considered faults like three-phase short-circuits with time of fault elimination equal to 0.15s (9 cycles considering a 60Hz operation), followed by the outage of the transmission line. The one-line diagram system is shown in Figure 1 (Appendix A). This system is composed by 10 synchronous machines, 73 transmission lines and 45 buses, based on the configuration of a southern Brazilian system. The desired outputs (critical clearing times) are data obtained of the transient stability studies (simulation) considering several levels of generation / load, corresponding to light load, base case, and two levels of heavy load, respectively 60, 80, 100 and 120% of base case (like as proposed in the reference [8]). In this case, the vectors ***P*** and ***Q*** can be substituted by a proportional parameter (p = 0.6, 0.8, 1 or 1.2), i.e., n is equal to ($1 + nb$). Table 2 shows the principal parameters referred to the used neural network and the training. Table 3 shows the comparative results by conventional BP and BP with fuzzy controller.

Table 2. Neural network specification.

Item	Value
Pattern vectors number	40
Number of layers	3
Number of neurons per layer	5-10-1
Tolerance	0.01
Training rate g*	5.5
Moment term h	0.8
Inclination of sigmoid function l	0.3
Parameter a	0.00071

Table 3. Comparative results.

Item	Conventional BP	BP with fuzzy controller
Number of cycles	5368	1334
Processing time (s)	14.12	4.28
Gain GC	1	4.02
Gain GT	1	3.29

Table 4 shows a comparative study of the results that were obtained by simulation and the use of neural network with fuzzy controller (Fuzzy BP). The transient stability system analysis is referred to the generation / load levels equal to 70, 90, 110 and 130% of the base case. The symbol $bk(ik\text{-}jk)$ presented in the first column of the Table 4 means the occurrence of a short circuit at bus bk with outage of transmission line between the nodes ik and jk. We observe at Table 4, that the results of neural network analysis are close to those obtained by simulation. It is verified that the analysis time, by neural network, is considerably less when compared to simulation. The program was processed in a Pentium II (450 MHz and 256 MB of RAM memory). The processing time is only referred to the BP algorithm execution, excluded the reading / output data operations.

Table 4. Comparative studies of the transient stability analysis.

Fault	Critical Clearing Time (seconds)							
	70 % of Base Case		90 % of Base Case		110 % of Base Case		130 % of Base Case	
	Simulation	Fuzzy BP	Simulation	Fuzzy BP	Simulation	Fuzzy BP	Simulation	Fuzzy BP
11(11–25)	0.5100	0.5128	0.3450	0.3398	0.2150	0.2144	0.1000	0.1200
15(14–15)	0.4200	0.4171	0.3550	0.3491	0.3000	0.2895	0.2250	0.2229
18(16–18)	0.2250	0.2199	0.1550	0.1493	0.0900	0.0917	0.0350	0.0502
18(18–44)	0.2250	0.2266	0.1550	0.1494	0.1000	0.0916	0.0400	0.0596
25(11–25)	0.5850	0.5968	0.4450	0.4489	0.2650	0.2706	0.1050	0.1227
25(25–26)	0.6650	0.6857	0.5750	0.5829	0.5050	0.4756	0.3700	0.3620
29(29–30)	0.4700	0.4724	0.3700	0.3825	0.2950	0.3068	0.2550	0.2330
33(33–36)	0.3100	0.3125	0.2400	0.2435	0.1800	0.1820	0.1250	0.1182
35(35–45)	0.4150	0.4158	0.3300	0.3284	0.2650	0.2622	0.2100	0.2072
39(39–40)	0.2700	0.2646	0.1950	0.1888	0.1350	0.1285	0.0900	0.0795

7. Conclusion

This work investigated the use of Artificial Neural Network for critical clearing times determination of faults elimination like short circuit in Electrical Power Systems. The employed Neural Network was that of multi-layers, non- recurrent with activation function as sigmoid. Then, the proposal of this work is to adjust the training rate *g* during the convergence process, to reduce its execution time. The results, considering a multi-machine system composed of 10 synchronous machines, 45 buses, and 73 transmission lines, can be satisfactory when compared to other methodology based on neural networks. In this work we consider several levels of generation / load, corresponding to light load, base case, and two levels of heavy load, i.e., are proportional levels. In sequence, are investigated other types of generation / load levels, no proportional that are more realist case, and better elaborated Electrical Power Systems dynamical models e.g., the complete model of Park ([1]).

8. References

[01] ANDERSON , P. M. and FOUAD, A. "Power System Control And Stability", 1977, IOWA State University Press, USA.

[02] ARABSHAHI, P.; CHOI, J. J.; MARKS II, R. J. and CAUDELL, T.P. "Fuzzy Parameter Adaptation in Optimization", IEEE Computational Science & Engineering, Spring 1996, pp. 57-65.

[03] CHO, H.S.; PARK, J.K. and KIM, G.W. "Power System Transient Stability Analysis Using Kohonen Neural Networks", Engineering Intelligent for Electrical Engineering and Communications, Vol. 7, No. 4, pp. 209-214, Dec. 1999.

[04] FONSECA, L. G. S. and DECKER, I. C. "Iterative Algorithm For Critical Energy Determination in Transient Stability of Power System", IFAC – Sym-posium Planning & Operation in Electric Energy Sys-tem, 1985, Rio de Janeiro - RJ, Brazil, pp. 483-489.

[05] KARTALOPOULOS, S.V. "Understanding Neural Net-woks and Fuzzy Logic", IEEE Press, New York, 1996.

[06] MINUSSI, C. R. and SILVEIRA, M. C. G. "Electric Power System Transient Stability By Neural Networks", 38 rd Midwest Symposium On Circuits And System, Rio de Janeiro-RJ, pp. 1305-1308, 1995.

[07] PAI, M.A. "Power System Stability – Analysis by the Direct Method of Lyapunov". North – Holland Publishing Company, 1981.

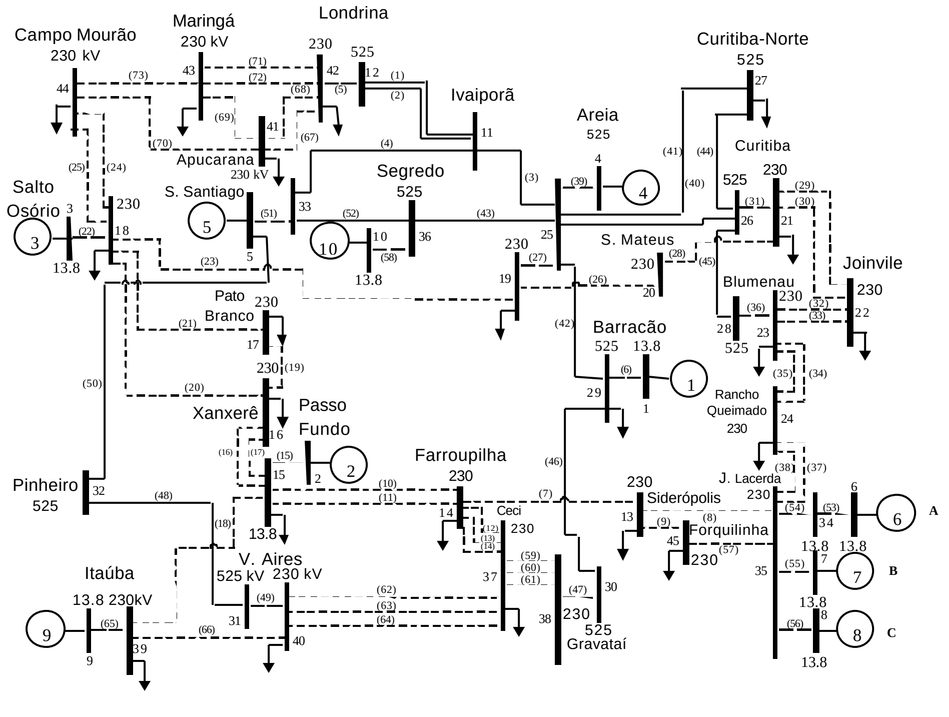
[08] PAO,Y.H. and SOBAJIC,D.J. "Combined Use of Un-supervised And supervised Learning For Dynamic Security Assessment", IEEE PICA-91, pp. 278 - 284.

[09] TERANO, T. ASAI, K. and SUGENO, M. "Fuzzy Systems Theory and Its Applications", Academic Press, USA, 1991.

[10] WIDROW, B.; LEHR, M.A. "30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation", Proceedings of the IEEE, Vol. 78, No. 9, Sep. 1990, pp. 1415-1442.

Appendix A

One-line Diagram of Power System



( ) Transmission line number.

Figure 1. Representation of test systems.