Automatica, Vol. 27, No. 1, pp. 115–134, 1991
Printed in Great Britain.

0005–1098/91 $3.00 + 0.00
Pergamon Press plc

Decision Trees and Transient Stability of Electric Power Systems*

L. WEHENKEL†‡ and M. PAVELLA†§

A general inductive inference method is proposed and applied to the automatic building of decision trees for the transient stability assessment of power systems. On the basis of large sets of simulations, the essential features of the method are analysed and illustrated.

Key Words—Artificial intelligence; pattern recognition; decision trees; electric power systems; data reduction and analysis; decision theory; stability; transient stability analysis, sensitivity and control.

Abstract—An inductive inference method for the automatic building of decision trees is investigated. Among its various tasks, the splitting and the stop splitting criteria successively applied to the nodes of a grown tree, are found to play a crucial role on its overall shape and performances. The application of this general method to transient stability is systematically explored. Parameters related to the stop splitting criterion, to the learning set and to the tree classes are thus considered, and their influence on the tree features is scrutinized. Evaluation criteria appropriate to assess accuracy are also compared. Various tradeoffs are further examined, such as complexity vs number of classes, or misclassification rate vs type of misclassification errors. Possible uses of the trees are also envisaged. Computational issues relating to the building and the use of trees are finally discussed.

1. INTRODUCTION

THE DECISION TREE methodology is nowadays recognized to be a generally nonparametric technique, able to produce classifiers in order to assess new, unseen situations, or to uncover the mechanisms driving a problem (Breiman *et al.*, 1984; Friedman, 1977; Kononenko *et al.*, 1984; Quinlan, 1986). The building of a decision tree is based on a learning set (LS), composed of a number of states together with their corresponding known classification. The building procedure starts at the top node of the tree with the entire LS, and progresses by recursively creating successor nodes, i.e. by splitting the LS into subsets of increasing classification purity. The procedure is stopped when all the newly created nodes are "terminal" ones, containing "pure enough" learning subsets. The ways of splitting the successive subsets, and even more of deciding when to stop splitting, are essential to the method. Often, the lack of general, efficient stop splitting methods is evaded by means of alternative procedures, such as first building a very large tree, then pruning it; they generally rely on empirical justifications applicable to particular cases, but without guarantee of effectiveness in other application domains. To remove this difficulty, a stop splitting criterion was developed on the basis of a general statistical hypothesis test (Wehenkel *et al.*, 1989a, b). It was designed independently of any specific application, then applied to power system transient stability.

Transient stability in general is concerned with the system ability to withstand severe contingencies. A possible measure of this is the *critical clearing time* (CCT), i.e. the maximum time that a contingency may remain without causing the irrevocable loss of machines' synchronism. To compute CCTs one may either use time-domain methods, which solve numerically the nonlinear differential equations describing the system motion, or direct methods which rely on the Liapunov criterion (Bergen, 1986; Ribbens-Pavella and Evans, 1985). Observe that the CCT is not appropriate enough for assessing transient stability, in that it carries partly useless and partly incomplete information; indeed, it merely provides a rather crude "yes-or-no" answer, whereas what really matters in practice is to assess "stability margins" and, if necessary, to suggest remedial actions (EPRI Project, 1987). Within the tree methodology, however, the CCT provides a handy means of classifying the states of a LS.

* Received 13 June, 1989; revised 10 January, 1990, received in final form 25 January, 1990. The original version of this paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor K-E. Arzen under the direction of Editor A. P. Sage.

† Department of Electrical Engineering, University of Liège, Institute Montefiore, B28, B 4000 Liège, Belgium.
‡ Research assistant F.N.R.S.
§ Author to whom all correspondence should be addressed.

The application of the tree methodology to transient stability was initially proposed in Wehenkel *et al.* (1986, 1987), then developed in Wehenkel *et al.* (1989a, b). The leading idea is to compress and organize the information about transient stability in the form of decision trees with the twofold objective: to classify new, unseen states, and to uncover the salient parameters driving the transient stability phenomena. Note that the foreseen advantages go beyond this objective: indeed, on-line means to compute stability margins and to infer control strategies were also suggested (Wehenkel *et al.*, 1988; Wehenkel, 1988). The first results obtained in a few practical examples were quite promising. The constructed trees exhibited nice features, notably with respect to accuracy and complexity. However, to make this method fully reliable and effective, a systematic exploration was necessary. This is a main purpose of the present paper.

Particular attention is paid to learning sets, "adequate" for building efficient decision trees; the way of generating their states in simulations and in real-life situations is discussed, and the influence of their number on the tree features is examined.

Another key issue concerns the splitting criterion. In Wehenkel *et al.* (1986, 1987), this consisted of a test applied to various "candidate attributes", chosen among static parameters of the power system, *a priori* likely to drive transient stability. Their influence on the tree structure is investigated below.

The cornerstone of the method is probably the stop splitting criterion (Wehenkel *et al.*, 1989a, b). Applied to transient stability, this criterion yielded simple, quite accurate trees in the few cases considered so far. In this paper, we determine its optimal parameters, and explore the accuracy and the complexity of the resulting trees.

The number of stability classes to be considered in the automatic building of trees is another question worth considering. Complexity, accuracy, misclassification rate and severity of misclassification errors are all interrelated features. Our purpose is to assess them, and to suggest possible practical applications, not necessarily to decide on a solution.

To conduct the above investigations we need appropriate tools for evaluating trees' accuracy. One of our first concerns will be to consider and compare *a priori* interesting evaluation criteria.

The ultimate goal for building trees is to use them. This topic is a whole research of its own. In this paper, we will merely indicate possible types of applications.

2. FUNDAMENTALS OF THE METHOD

2.1. *General framework and basic notation*

The automatic building of decision trees by the proposed inductive inference method implies the existence of a learning set, i.e. of a number, say N, of preclassified *states*. Without loss of generality, we will assume that each state is characterized by a certain number, say n, of ordered numerical *attributes* (the same number for each state), and that the N states are classified into two classes only $\{+, -\}$. The generalization to more than two-class classifications will be considered in Section 2.10; the extension to categorical attributes in addition to the ordered ones would be quite straightforward.

In the sequel, a learning set (LS) will be defined by:

$$LS \triangleq \{(\mathbf{v}_1, c_1), (\mathbf{v}_2, c_2), \ldots, (\mathbf{v}_N, c_N)\} \quad (1)$$

where the components of vector $\mathbf{v}_k$

$$\mathbf{v}_k = (v_{1k}, v_{2k}, \ldots, v_{nk})^T \quad (2)$$

represent the attribute values of the state $\mathbf{s}_k$, which is characterized by its n attributes:

$$\mathbf{s}_k = [a_1 = v_{1k}] \cap [a_2 = v_{2k}] \cap \cdots \cap [a_n = v_{nk}], \quad (3)$$

and where

$$c_k \in \{+, -\}. \quad (4)$$

Note that the preclassified learning set is considered to be a statistical sample of size N, drawn from the population of possible states.

The test set (TS), is defined as a similar, but independent sample of size M:

$$TS \triangleq \{(\mathbf{v}_{N+1}, c_{N+1}), (\mathbf{v}_{N+2}, c_{N+2}), \ldots, (\mathbf{v}_{N+M}, c_{N+M})\}. \quad (5)$$

It will be used for the purpose of evaluating the performance of a decision tree with respect to unseen states (see Section 2.8).

2.2. *Decision trees (DTs)*

A DT is a tree structured upside down. It is composed of *test* and *terminal* nodes, starting at the *top node* (or *root*) and progressing down to the terminal ones. Each test node is associated with a *test on the attribute values of the states*, to each possible outcome of which corresponds a successor node. The terminal nodes carry the information required to classify the states. Such a DT is portrayed in Fig. 1, built for an example treated in Section 4 in the context of transient stability. Note that in this case, where only numerical attributes are considered, the tests are dichotomic, and the resulting DT is binary: each test node is split into *two* successors.

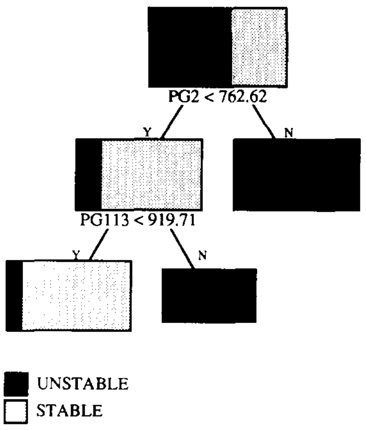

FIG. 1. A biclass tree for contingency #2. $N = 500$, $\alpha = 0.0001$.

A convenient way to define a DT is to describe the way it is used for classifying a state of *a priori* unknown classification on the basis of its known attributes. This classification is achieved by applying sequentially the test at the test nodes, beginning at the root of the tree, and systematically passing the state to the successor appropriate to the outcome of the test, until a terminal node is finally reached; the state is classified accordingly.

The above description provides an interesting, geometric interpretation of a tree procedure (Breiman *et al.*, 1984): it recursively partitions the attribute space into hyperboxes, such that the population within each box becomes more and more class homogeneous. This in turn allows one to view the classification of a tree as the partitioning of this hyperspace into two regions, corresponding to the two classes. Each class is composed of the union of the elementary boxes corresponding to its terminal nodes. Figure 2 illustrates the geometric representation of the tree of Fig. 1. (Actually, according to this tree structure, the states belonging to the region labelled "stable" have a small chance (13%) to be unstable.)

2.3. *Automatic construction of DTs*

For a given LS, our purpose is to build a near optimal DT, in the sense that it realizes a good tradeoff between complexity and accuracy, i.e. between total number of nodes and classification

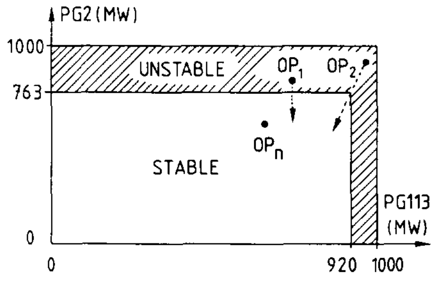

FIG. 2. Geometric representation of the tree of Fig. 1.

ability. Usual ways of estimating accuracy are specified in Section 2.8; anticipating, observe that, intuitively, assessing trees' ability to correctly classify states should rely on test states rather than on states belonging to LS itself.

The ultimate objective of the building procedure is to: (i) select the relevant attributes among the candidate ones (generally the number of retained attributes is much lower than n, the total number of candidate attributes); (ii) build the decision tree on the basis of these relevant attributes. This building procedure is described below (Wehenkel *et al.*, 1989a).

- Starting at the root of the tree, with the list of candidate attributes and with the whole LS, the learning states are analysed in order to select a test which allows a maximum increase in purity or, equivalently, which provides a maximum amount of information about their classification. The selection proceeds in two steps:
  (i) for each attribute, say a_i, it finds the optimal test on its values, by scanning the values of this candidate attribute for the different learning states; in our case of ordered numerical attributes, this step provides an optimal threshold value v_{i*} and defines the test
  $$a_i < v_{i*}? \qquad (6)$$
  (ii) among the different candidate attributes, it chooses the best one, a_*, along with its optimal value, v_{**}, to split the node.

  In short, step (ii) defines the optimal attribute, step (i) its optimal threshold value.
- The selected test is applied to the learning set of the node and splits it into two subsets, corresponding to the two successors of the node. Starting with the root of the tree and the entire LS, the two subsets
  $$LS_1 \triangleq \{\mathbf{v}_k \in LS \mid a_* < v_{**}\}$$
  $$LS_2 \triangleq \{\mathbf{v}_k \in LS \mid a_* \geq v_{**}\},$$
  correspond to the two successors of the root.
- The successors are labeled terminal or not on the basis of the *stop splitting criterion* described below.
- For the nonterminal nodes, the overall procedure is called recursively in order to build the corresponding subtrees.
- For the terminal nodes, the class probabilities p_+ and p_- are estimated on the basis of the corresponding subset of learning states there stored, and the class label of the majority class is attached.

Obviously, the crux of the entire construction of a DT lies in the selection of the splits and the of a DT lies in the selection of the splits and the decision whether to declare a node terminal or to continue splitting. These two questions are examined below, after the introduction of some necessary notions of (im)purity and information, drawn from the information theory.

2.4. *Mathematical formulation*

2.4.1. *Definitions.* Denoting by:

S the subset of all possible states, corresponding to some node of a DT, i.e. directed to that node by the classification procedure;

$p(+ \mid \mathbf{S})$ (resp. $p(- \mid \mathbf{S})$) the probability of a random state drawn from **S** to belong to the class + (resp. −);

T a dichotomic (candidate) test [e.g. see (6)] on some attribute's values of the states;

$\mathbf{S}_y$ (resp. $\mathbf{S}_n$) the subsets of **S** of states yielding the answer "YES" (resp. "NO") to the test T; these sets would correspond to the successors of the node split on the basis of T;

$p(\mathbf{S}_y \mid \mathbf{S})$ (resp. $p(\mathbf{S}_n \mid \mathbf{S})$) the probabilities of the outcomes "YES" and "NO" in **S**; one defines the following measures.

The *prior* "classification entropy" of **S**

$$H_C(\mathbf{S}) \triangleq -[p(+ \mid \mathbf{S}) \log_2 (p(+ \mid \mathbf{S})) + p(- \mid \mathbf{S}) \log_2 (p(- \mid \mathbf{S}))]. \quad (7)$$

$H_C(\mathbf{S})$ is a measure of the impurity or the uncertainty of the classification of a state of **S**; $H_C(\mathbf{S}) = 0$ corresponds to a perfectly pure ($p(+ \mid \mathbf{S}) = 1$ or 0) subset, whereas $H_C(\mathbf{S}) = 1$ corresponds to $p(+ \mid \mathbf{S}) = p(- \mid \mathbf{S}) = \frac{1}{2}$, i.e. maximal uncertainty.

The entropy of **S** with respect to the test T

$$H_T(\mathbf{S}) \triangleq -[p(\mathbf{S}_y \mid \mathbf{S}) \log_2 (p(\mathbf{S}_y \mid \mathbf{S})) + p(\mathbf{S}_n \mid \mathbf{S}) \log_2 (p(\mathbf{S}_n \mid \mathbf{S}))]. \quad (8)$$

$H_T(\mathbf{S})$ is a measure of the uncertainty of the outcome of T in **S**; with respect to the outcome of T, it has similar properties to H_C with respect to the classification.

The mean *posterior* "classification entropy" of **S** given the outcome of T

$$H_{C|T}(\mathbf{S}) \triangleq p(\mathbf{S}_y \mid \mathbf{S}) H_C(\mathbf{S}_y) + p(\mathbf{S}_n \mid \mathbf{S}) H_C(\mathbf{S}_n). \quad (9)$$

$H_{C|T}(\mathbf{S})$ is a measure of the residual impurity if **S** is split into $\mathbf{S}_y$ and $\mathbf{S}_n$ according to the outcomes of test T.

The "information" provided by T on the classification of **S**

$$I_C^T(\mathbf{S}) \triangleq H_C(\mathbf{S}) - H_{C|T}(\mathbf{S}); \quad (10)$$

$I_C^T(\mathbf{S})$ is a measure of the ability of T to produce pure successors.

The normalized information gain of T

$$C_C^T(\mathbf{S}) \triangleq \frac{2 I_C^T(\mathbf{S})}{H_C(\mathbf{S}) + H_T(\mathbf{S})}. \quad (11)$$

Remark. The measures defined in (7)–(10) are expressed in bits whereas that in (11) is dimensionless.

2.4.2. *Interpretation.* It is possible to give a qualitative interpretation of the relation between $C_C^T(\mathbf{S})$ and the classification ability of the test T. To this, first observe that the following inequalities can be easily shown:

$$0 \le H_C(\mathbf{S}), H_T(\mathbf{S}), H_{C|T}(\mathbf{S}), I_C^T(\mathbf{S}), C_C^T(\mathbf{S}) \le 1 \quad (12)$$

$$H_{C|T}(\mathbf{S}), I_C^T(\mathbf{S}) \le H_C(\mathbf{S}). \quad (13)$$

Then consider the two following extreme cases:

1. The outcome {"YES", "NO"} of T and the classification $\{+, -\}$ are statistically independent, i.e. the test T provides no information at all about the classification. In this case

$$p(+ \mid \mathbf{S}_y) = p(+ \mid \mathbf{S}_n) = p(+ \mid \mathbf{S}) \quad \text{and} \quad p(- \mid \mathbf{S}_y) = p(- \mid \mathbf{S}_n) = p(- \mid \mathbf{S}). \quad (14)$$

Thus, $H_C(\mathbf{S}_y) = H_C(\mathbf{S}_n) = H_C(\mathbf{S})$ and according to (9), $H_{C|T}(\mathbf{S}) = H_C(\mathbf{S})$. In addition, by virtue of (10) and (11),

$$I_C^T(\mathbf{S}) = C_C^T(\mathbf{S}) = 0.$$

2. The outcome of T provides pure classified subsets $\mathbf{S}_y$ and $\mathbf{S}_n$. Then necessarily $H_C(\mathbf{S}_y) = H_C(\mathbf{S}_n) = 0$. According to (9), this implies that $H_{C|T}(\mathbf{S}) = 0$ and $I_C^T(\mathbf{S}) = H_C(\mathbf{S})$. Moreover, in this case the probability $p(\mathbf{S}_y \mid \mathbf{S})$ (resp. $p(\mathbf{S}_n \mid \mathbf{S})$) will be equal to either $p(+ \mid \mathbf{S})$ (resp. $p(- \mid \mathbf{S})$) or $p(- \mid \mathbf{S})$ (resp. $p(+ \mid \mathbf{S})$), and thus $H_T(\mathbf{S}) = H_C(\mathbf{S})$ and

$$C_C^T(\mathbf{S}) = 1.$$

From the above it follows that the higher the value of $C_C^T(\mathbf{S})$, the more interesting the test T for splitting the node corresponding to **S**. This is exploited in Section 2.5 to define the optimal splitting rule, used in the building of DTs.

2.4.3. *Estimation on the basis of the sample of learning states.* The information and entropy measures defined in (7)–(11) cannot be computed directly since the probabilities involved are generally unknown. Therefore, we use the learning set as a statistical sample and estimate the probabilities by their empirical values computed in the LS. More precisely, the set **S**, corresponding to some node of the DT, is replaced by the *subset* of learning states (i.e. $\mathbf{S} \cap \mathrm{LS}$) corresponding to this node. The computation of $p(+ \mid \mathbf{S})$, $p(- \mid \mathbf{S})$, $p(\mathbf{S}_y \mid \mathbf{S})$ and $p(\mathbf{S}_n \mid \mathbf{S})$ of this subset is then straightforward, since the classification and attribute values of the learning states are known.

It should be noted that, although the estimates of $p(+ \mid \mathbf{S})$, $p(- \mid \mathbf{S})$, $p(\mathbf{S}_y \mid \mathbf{S})$ and $p(\mathbf{S}_n \mid \mathbf{S})$ are generally unbiased, their substitution in (7)–(11) provides rather optimistically biased information measures, thus overestimating the actual "goodness" of the test T. Fortunately, the amount of bias is inversely proportional to the sample size (the number l of states in $\mathrm{S} \cap \mathrm{LS}$) and the measures may still be used for comparison purposes, e.g. in order to select an "optimal" test for splitting the node.

For example, it can be shown (Kvålseth, 1987) that for an actual value of $I_C^T(\mathrm{S}) = 0$ (no correlation), its estimate

$$\hat{I} = I_C^T(\mathrm{S} \cap \mathrm{LS}) \tag{15}$$

has a χ-square like distribution, the expectation (or bias) of which is positive and inversely proportional to the number l of states in $\mathrm{S} \cap \mathrm{LS}$. This property is exploited below, in the stop splitting criterion.

In the sequel, to emphasize the difference between the purity measures and their estimates, we will call the latter "apparent" as opposed to the "real" unknown values.

2.5. *On the optimal splitting criterion*

Obviously, the splits at the test nodes should be selected so as to avoid to the extent possible the appearing of deadends (i.e. of impure terminal nodes, see the definition given below in Section 2.6), and to obtain the desired degree of accuracy. This is done in a more or less heuristic (not necessarily optimal) fashion: the best test (defined by the optimal attribute together with its optimal threshold value, see Section 2.3), is considered to be the one which separates at most the states of the two classes in the local learning subset. This strategy amounts to selecting the split which yields the purest direct successors, or maximizes the apparent normalized information gain $C_C^T(\mathrm{S} \cap \mathrm{LS})$ defined in (11). In that sense, it may be considered to be *locally*, rather than *globally* optimal.

2.6. *Stop splitting method*

Many methods were proposed. For example, ID3 (Quinlan, 1986) stops splitting at a node only if the corresponding learning subset is completely included in one of the classes of the goal partition. Unluckily, in many situations, as shown by our experience, this strategy tends to build overly complex DTs, most of the terminal nodes of which contain only a very small and unrepresentative sample of learning states. They perform generally badly with respect to unseen situations and are unable to indicate in a reliable way the inherent relationship between the attributes and the goal classification. To circumvent this difficulty we propose a more conservative criterion, which stops splitting a node if one of the following two conditions is met:

1. The local subset of learning states is sufficiently class pure. Such a terminal node will be called a *leaf* in the sequel. The degree of class purity required for leaves is a parameter of the algorithm and fixes the amount of detail we want the DT to express.

Note. Actually, the degree of residual "impurity" may be specified by H_m, the maximal residual entropy [see (7)]. The entropy of the learning subset relative to a node, is inversely proportional to its purity. Thus, in terms of entropy, a node will be a leaf only if its entropy is lower than H_m. In the practical investigations reported in this paper, a constant value of $H_\mathrm{m} = 0.1$ bits was used. This very low value amounts to building DTs as detailed as possible. Higher values could be of interest if one wanted to build simpler "first guess" DTs, for data exploration purposes.

2. There is no possibility to enhance the DT accuracy in a statistically significant way by splitting this node further. In other words, given the optimal dichotomic split for this node, there is not enough evidence in the local learning subset, that this split would actually improve the real performance of the DT. Such a terminal node is called a *deadend* in the sequel. This second criterion prevents the building of unnecessarily complex DTs. It is formulated as a statistical hypothesis test:

Given the local subset of learning states and the optimal split, can we accept the hypothesis that the apparent increase in accuracy, as measured in this subset, provided by the split, is a purely random effect?*

In quantitative terms, the statistic $\hat{A} \triangleq cl\hat{I}$ (where l is the size of the local learning subset, c a constant and $\hat{I}$ the *apparent* increase in purity provided by the optimal split) is distributed according to a χ-square law (with one degree of freedom) under the hypothesis of no *real* increase in purity. Hence, if we fix the α-risk of not detecting these situations to a given value, testing the value of $\hat{A}$ against the threshold $\hat{A}_{cr}$ such that $\mathrm{Prob}(\hat{A} \geq \hat{A}_{cr}) = \alpha$ allows one to detect the cases where the apparent increase in accuracy is a random effect, with a probability of $1 - \alpha$. Figure 3 sketches such χ-square probability density functions.

* Apparent as opposed to real increase in accuracy, means the increase measured in the LS as opposed to the increase measured for unseen states.

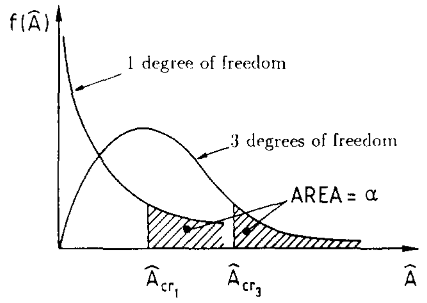

FIG. 3. χ-square probability density functions of $\hat{A}$.

Thus, the α-risk of the hypothesis test fixes the amount of evidence we require at each node in order to split it; the answer depends on the value of α, the size of the local learning set, and the amount of apparent accuracy improvement provided by the test. The question of "how much evidence should be required to allow the splitting of a node", is related to the degree of representativity we impute to the learning set and the risk that this degree is overestimated. Therefore, the degree of cautiousness of the procedure is fixed by the user via the selection of the value of α. This value ranges from 1 (the criterion has no effect on the splitting procedure anymore, and the tree grows according to the above condition 1) to zero (no growth is allowed, the tree reduces to its root).

The value of α has indeed drastic effects on the resulting trees characteristics as illustrated by the following example (borrowed from Section 4, below). Four DTs were grown for α assuming the values 1.0, 0.1, 0.01 and 0.0001; they were built on the basis of a LS composed of 500 states and evaluated on the basis of an independent test set of 1500 other states. Table 1 reports their complexity and accuracy, i.e. the number of their nodes and the percentage of correctly classified test states. These quite impressive figures illustrate that the larger trees are less accurate than the smaller ones and on the other hand that the appropriate selection of α allows the building of small, yet better DTs (see also the discussion of Section 2.9 and the simulation results of Section 4.3.2).

2.7. *Comparison with the ID3 method*

The proposed inductive inference method originates from the ID3 algorithm, from which, however, it departs in some essential respects.

TABLE 1

α	1.00	0.1	0.01	0.0001
Complexity	63	55	7	5
Accuracy	88.1	88.5	91.2	91.2

Below, we collect the main differences between the two methods.

Application domain. ID3 was initially intended to handle mainly *symbolic* and *deterministic* learning problems characterized by very large (almost complete) learning sets composed of objects described by discrete (or qualitative) attributes only. Thus, it was essentially designed to handle large sets of data, in order to *compress* rather than extrapolate their information (Quinlan, 1984). On the contrary, the proposed method is especially tuned to handle mainly *numeric* and *nondeterministic* problems, where the learning set has to be *generalized* in an appropriate fashion. The method is general enough to handle at the same time numeric and qualitative attributes and can be tuned to the degree of "representativeness" of the learning set, via the appropriate choice of the threshold α (see Section 4.3 below).

Stop splitting criterion. ID3's stop splitting criterion amounts to building DTs which classify the learning set as correctly as possible, which is indeed the best approach if the latter is almost complete. The proposed method, on the other hand, stops splitting earlier, as soon as the statistical hypothesis test alows to conclude that no *significant* improvement of the tree's accuracy would be achieved by developing the node anymore. This is quite different, and enables the method to take the best advantage of learning sets which are only partly representative of the possible objects. This is further discussed in Section 2.9 below.

Optimal splitting criterion. ID3 considers the best test to be the one providing the largest apparent information gain $\hat{I}_C^T$. It has been found that this measure is biased in the favour of those having the largest number of successors, especially in the context of randomness (Quinlan, 1986). Note that this is supported by the fact that the number of degrees of freedom of the χ-square distribution (and thus its bias) increases linearly with the number of successors of a split. The *normalized* correlation measure $\hat{C}_C^T$ evaluates the tests more objectively since the number of succesors is compensated by the term $\hat{H}_T$ in its denominator.

Strategy. ID3's strategy, embodied by its optimal and stop splitting criteria, amounts to building trees which maximize the *apparent reliability*, regardless of their complexity. On the other hand, the strategy of the proposed method is to build DTs maximizing a measure of *quality* which realizes a compromise between *apparent reliability* and *complexity*. The latter is more effective in the context of incomplete and random data (Wehenkel, 1990).

2.8. *Evaluation criteria*

The criteria described below (e.g. see Toussaint, 1974) allow assessing the accuracy of a DT in general, or equivalently, its misclassification rate. Observe that these are only estimates, since in real world problems it is seldom possible to scan all the possible states, even in deterministic type problems.

Resubstitution estimate, R^{ls} obtained by considering all the states belonging to the LS, dropping them through the DT, and computing the ratio of misclassified over the total number (N) of states;

Test sample estimate, R^{ts} obtained by considering all the states belonging to the TS, dropping them through the DT, and computing the ratio of misclassified over the total number (M) of states;

Cross-validation estimate, R^{cv} obtained by: (i) dividing the total number of learning states into, say V, equally sized subsets, using one of them as the test set and the union of the remaining $V-1$ as a learning set; (ii) building a DT based on this learning set and assessing its accuracy on the basis of the test set; (iii) repeating this procedure V times by considering successively each of the V subsets as test set; (iv) taking the average of the V individual estimates as the final accuracy. The validity of the procedure implies that V is quite large (>10), so that the DTs constructed on the basis of the $(V-1)$ subsets are (almost) identical with the initial DT, constructed on the basis of the total number of states (V subsets). For $V = N$ this is the so-called "leave-one-out" estimate.

Remark. The above defined measures can be considered as more or less accurate estimates of the true probability of misclassifying a new state. They have the following characteristics.

R^{ls} is easy to compute and requires no additional samples but is generally optimistically biased, underestimating the real probability of error. Intuitively, for fixed N, the larger the DT, the higher the correlation between the DT and the LS, and the more biased R^{ls}.

R^{ts} is the most reliable and unbiased estimate and is easy to compute. Moreover, if M is sufficiently large its variance is small. Its main drawback is that it requires additional states in sufficiently large number, which cannot be used in the learning set. Another advantage of R^{ts} is that one can easily estimate its variance and therefrom compute confidence intervals. In the sequel we will consider this estimate as the benchmark.

R^{cv} combines advantages of the two preceding measures, since it is generally less biased than R^{ls} and does not require additional states. On the other hand, its variance can be very large, and certainly depends strongly on the characteristics of the problem at hand. In practical situations, when the number of samples is limited, it is an interesting alternative to R^{ts}. Notice also that the computational burden required for R^{cv} can be very heavy, since it needs the building of V additional DTs.

The above considerations are illustrated in Section 4, on a real world example.

2.9. *On the right size of trees*

The stop splitting method introduces the statistical threshold parameter α. Varying its value allows to modify the size of a tree. But how to choose the right tree size? The choice should be guided by the observation that too large a DT will yield a higher misclassification rate and hide relevant relationships, whereas too small a tree will not make use of some of the information available in the LS. Indeed, the terminal nodes of a too small DT have not been expanded enough and this prevents from getting the purer subsets and the corresponding insight about the role that the attributes would have played in this expansion; a too large DT, on the other hand, results from the splitting of statistically unrepresentative subsets; therefore, it is likely to cause an increase in the misclassification rate when classifying states not belonging to the LS. Stated otherwise, a tradeoff appears between the two following sources of misclassification: *bias* (overlooking significant information in the LS) and *variance* (badly interpreting the randomness in the LS): too large a tree will suffer from variance whereas too small a tree will present bias.

2.10. *Building multiclass DTs*

The above description of the inductive inference method made in the case of two classes $\{+, -\}$, is general enough to handle (at least in principle) an arbitrary number, say m of classes, provided that m remains negligible with respect to the size N of the LS.

Indeed, on one hand the information theoretic purity measures defined in (7)–(11) may be generalized to m classes; on the other hand the statistical hypothesis test remains applicable, provided that $m-1$ degrees of freedom are used (instead of 1 in the two-class (or biclass) case) for the χ-square law.

Under these conditions, the method will build

DTs classifying directly the states into one of the m specified classes.

Another indirect possibility would be to build $m-1$ biclass DTs and to combine them in order to obtain the m-class classification.

In the investigations of Section 4 we use the first, direct approach. The obtained DTs are simpler and easier to interpret. Moreover, preliminary investigations indicate that they are at least as good, sometimes better, from the accuracy viewpoint, than the indirect multi-biclass trees.

3. DECISION TREE BASED TRANSIENT STABILITY APPROACH

Two main conjectures underlie the application of the tree method to transient stability assessment. First, transient stability is strongly dependent upon the contingency type and location; hence the idea of building a tree per contingency. Second, transient stability is a quite localized phenomenon, and is driven by a few number of the system parameters; hence the idea of proposing candidate attributes selected among the parameters of the system in its steady state condition.

These generally well-accepted conjectures, have also been verified in the few cases treated by the tree methodology (Wehenkel *et al.*, 1989a, b). The case study of the next section attempts to further validate them by means of exhaustive simulations. It also provides answers to questions raised by the overall *decision tree transient stability* (DTTS) method, whose principle is recalled below.

3.1. *Principle of the DTTS approach and related questions*

This may be formulated as follows: for each preassigned contingency, build up off-line a DT on the basis of a learning set and of candidate attributes. This tree is then used on-line to classify new, unseen states in as many classes as desired; for example, in a biclass tree, a given state would be classified as either stable or unstable, whereas in a three-class tree, the same state would be declared stable, fairly stable, or unstable.

Below, we identify questions relevant to this definition and suggest answers, often anticipating the results of Section 4.

3.1.1. *Questions about the LS.*

(i) How to obtain the learning states;
(ii) How to classify them;
(iii) How many states should be used for the efficient construction of a DT;
(iv) Whether the "right" size of LS should be dependent upon the size of the power system.

Tentative answers.

(i) Either by considering plausible scenarios and running a load flow program to get the corresponding operating states, or by using past records of the system real life;
(ii) According to their CCT values, computed via a time-domain or a direct method as appropriate;
(iii) This question is explored in the next section;
(iv) The answer to this question is postponed until Section 5.

3.1.2. *Questions about the list of candidate attributes.*

(i) What kind of candidate attributes to select for test (6);
(ii) How many;
(iii) What would happen if the actually most relevant attribute were for any reason masked, i.e. discarded from the list;
(iv) What if additional relevant attributes are further considered.

Tentative answers.

(i) Parameters of the system in its steady state, pre-contingency, operation;
(ii) When no prior knowledge of the system can guide the selection, it is advisable to consider as many as possible candidates of the type just suggested, confined in a relatively restricted area surrounding the contingency location; note that the increase in computing cost is insignificant (see below);
(iii) It would generally lead to somewhat more complex, yet quite accurate DTs, provided that other relevant attributes are still present in the list;
(iv) See results of Section 4.3.4.

3.1.3. *Questions relating to the number of classification patterns.*

(i) Which are the main differences between trees of small and large number of classes;
(ii) For a given misclassification rate, which type of trees provides narrower misclassification error.

Tentative answers

(i) One can reasonably conjecture that the smaller the number of classes in a tree, the less complex and more accurate this tree;
(ii) Provided that the inductive inference method is correctly developed, it is normal to think that the larger the number of classes in a tree, the less severe the misclassification

errors; indeed, in a well designed tree, misclassification errors will result in adjacent classes; it is therefore less severe to declare fairly stable a state which actually is stable (three-class tree), than to declare it unstable instead of stable (two-class tree) (see Section 4.3.5).

3.1.4. *Questions relating to computational aspects.*

(i) Which is the most expensive task of the DTTS approach;
(ii) How does the number of candidate attributes affect the computing time of a DT;
(iii) How does the number N of learning states affect the computing time of a DT;
(iv) How "expensive" is the storage of DTs;
(v) How "expensive" is their use.

Tentative answers.

(i) The generation of a LS is undoubtedly the most demanding task; and the more refined the system modelling, the more expensive the task;
(ii) According to the splitting criterion procedure described in Section 2.3, the time required by a DT building varies linearly with the number of candidate attributes;
(iii) In terms of the size of the learning set, the computing time is upper-bounded by—and generally much lower than—$N \log N$, i.e. the time required to sort the LS according to the values of the n candidate attributes;
(iv) The storage of a DT is generally extremely inexpensive; indeed, it is proportional to the number of its nodes, which is found to be quite small (see next section); to fix ideas, a compiled LISP version of a DT-classifier composed of 25 nodes (which can be considered as an upper bound for the DTTS method) takes about 600 machine instructions, meaning that in the context of modern computer architectures thousands of DTs can be stored simultaneously into main memory;
(v) The mean classification time of a state by means of a DT is almost negligible (about 0.6 ms for a DT comprising 25 nodes).

3.2. *Possible types of uses of the DTs*

Various uses can be inferred by following the pattern of the general DT methodology (e.g. see Section 2.2); many others are specific to the DTTS approach.

The use which immediately comes to mind is the classification of an operating state of *a priori* unknown degree of stability with respect to a given contingency: considering the appropriate DT, and applying to this state the test (6), one successively directs it through the various nodes of the tree, until reaching a terminal node; the state is classified accordingly. One could object that the information thus obtained is less refined than the CCTs used to classify the states of the LS; this however is not surprising, since the precise CCT values of the learning states are not fully exploited during the tree building. Wehenkel *et al.* (1988) and Wehenkel (1988) proposed a means to approximately estimate the CCT of the state one seeks to classify, by using the notion of "distance to a class", whose definition takes into account the CCTs. This distance, inferred from the geometric interpretation of a DT outlined in Section 2.2, is schematically illustrated in Fig. 2. One can see that the *operating* state (or *point*, OP) labelled OP_n is stable, whereas the state OP_1 is unstable and OP_2 is very unstable. Moreover, it shows that to (re)inforce stability, one should suitably direct the OP in the attribute space, i.e. suitably modify the values of its relevant attributes.

The above short description suggests that the DTTS approach can provide three types of transient stability assessment: analysis, directly linked to the classification of an OP, sensitivity analysis, linked to the "distance of an OP to a class", and control, linked to the way one can act to modify such a distance. Moreover, since the involved computations are likely to be extremely fast, several real-time strategies may be conceived. For example, one may proceed in a way similar to the standard "contingency evaluation" of steady state security assessment, *viz.*: draw up a contingency list, and build off-line the corresponding DTs; then, in real-time, scan the list, focus on those contingencies which are likely to create problems, and if necessary propose corrective actions. Of course, care should be taken so as to avoid incompatible actions. But for the time being this question is beyond our scope.

Finally, observe that the DTs provide also a clear, straightforward insight into the complex mechanism of transient stability. This is not the least interesting contribution of the DTTS approach.

4. CASE STUDY

4.1. *General description*

The investigations conducted in this section aim to answer the questions previously raised, to assess the salient features of the DTTS approach and to test the conjectures underlying it.

The power system we have chosen is small enough to avoid unnecessarily bulky computations, but large enough to draw realistic conclusions. It is composed of 31 machines, 128 buses and 253 lines; its total generation power in the base case amounts to 39,000 MW. This system is described in Lee, (1972), and sketched in Fig. 4. (The rationale of its decomposition is discussed in Section 4.2.1.)

In this first set of large-scale simulations we adopted the standard simplified system modelling, i.e. constant electromotive force behind transient reactance for each machine and constant impedance for each load.

To assess the DTTS method as objectively as possible, we have generated 2000 operating points (OPs). Two sets of simulations were considered, corresponding to very severe contingencies, consisting of three-phase short-circuits (3ϕSCs) applied at generator buses (one at a time). The first set concerns thorough investigations carried out with three such stability scenarios, of 3ϕSCs applied at buses #2, 21, and 49, arbitrarily chosen.* This implied the computation of 6000 CCT values. Part of the OPs along with their classification have been used in the LS, the other part composing the TS. A large number of DTs have been built for the above three contingencies, for various sizes of LSs, varying values of the threshold α, and three different classification patterns, namely two-, three-, and four-class classifications. In the two-class case, one distinguishes stable and unstable OPs, depending upon whether their CCT is above or below a certain threshold CCT value; in the three-class case one distinguishes stable, fairly stable and unstable OPs (two threshold CCT values are used); in the four-class case one distinguishes very stable, stable, unstable and very unstable OPs (three threshold values). The threshold CCT values are *a priori* chosen rather arbitrarily, but so as to avoid a too important imbalance between the populations of OPs belonging to the various classes.

The second set of simulations is described in Section 4.4.

4.2. *Constitution of a data base*

The data base was randomly generated on the basis of plausible scenarios, corresponding to various topologies, load levels, load distributions and generation dispatches. Hereafter we describe the way used to generate them, to analyse them from the transient stability point of view, and to build the attribute files.

4.2.1. *Random generation of OPs.* To generate these various states, we grouped the nodes of the power system into five internal zones, and one external zone composed of the boundary nodes of the system and its external equivalents (Fig. 4). The internal zones were defined empirically, on the basis of the "electrical distances" (number and length of lines) connecting their nodes. The OPs composing the data base were generated randomly according to the following independent steps.

1. Topology: it is selected by considering base cases (with probability 0.5) and single outage of a generator, a load or a shunt reactor (each with probability 0.08), a single line (with probability 0.16), two lines (with probability 0.1). The outaged element is selected randomly among all the elements of the power system.
2. Active load level: the total load level is defined according to a Gaussian distribution (with $\mu = 32{,}000$ MW and $\sigma = 9000$ MW).
3. Distribution of the active load: the total load is first distributed among the six zones according to the random selection of participation factors (see below), then among the load buses of each zone homothetically with respect to their base case values. The reactive load of each bus is adjusted according to the local base case power factor. This results in a very strong correlation of the loads of a same zone, and a quite weak one among loads of different zones.
4. Distribution of the active generation: in a similar fashion, the total generation is first distributed among the zones according to randomly selected participation factors, then among the generators of each zone according to a second selection of participation factors. Thus, neighbour generators are less correlated than neighbour loads. The reactive generations are obtained by a load flow calculation. To avoid overloads, the final active generation of each generator is constrained to 90% of its nominal power.

Note. In order to avoid overloading or underloading the slack generator, the total generation is defined as the total active load plus a polynomial approximation of the active network losses of form:

$$\text{Network Losses (MW)} \approx 1336 - 255P_l + 19P_l^2 - 0.7P_l^3 + 0.012P_l^4 - 0.00007P_l^5$$

where P_l denotes the total active load of the OP, in GW, and where the coefficients were determined by a least squares estimation on the basis of 11 OPs of different load levels for which the network losses were computed by a load flow program.

* In the sequel, a contingency will often be specified by merely the number of the generator bus at which it is supposed to apply; e.g. contingency (or fault) #2, #21, #49.

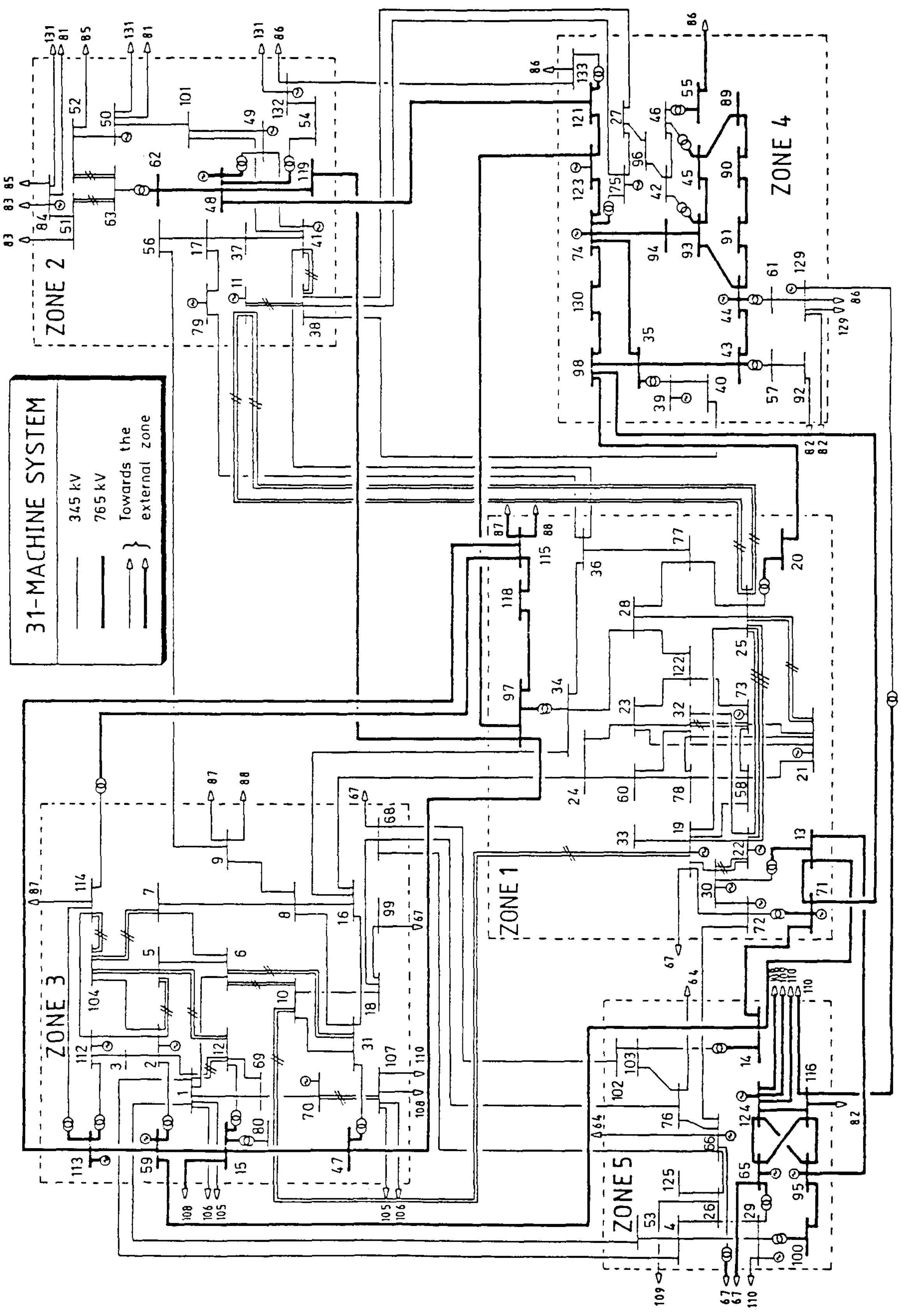

FIG. 4. One-line diagram of the 31-machine test system and its decomposition.

5. Load flow calculation: to check the feasibility of an operating point and compute its state vector, it is fed into the load flow program, and accepted if the latter converges properly. (About 90% of the states were accepted.) (Remember that the power system has 128 buses and hence each state vector has $2 \times 128 - 1 = 255$ components.)

Participation factors. To distribute a given amount of active (generation or load) power (say P_{tot}) on a number, say E, of "elements" (zones, loads or generators) we use the following procedure.

Let P_i^N, $(i = 1, \ldots, E)$ be the "nominal" power of the elements (for zonal loads we use the base case total load of the zone, for generators their nominal power and for zonal generations the sum of the nominal generations of their generators), and λ_i, $(i = 1, \ldots, E)$, positive coefficients selected randomly, according to the procedure described below. The individual participation of each element is defined by:

$$P_i \triangleq \frac{\lambda_i P_i^N}{\sum_{j=1}^{E} \lambda_j P_j^N} P_{\text{tot}}. \tag{16}$$

Thus, the λ_i coefficients act as a distortion with respect to the homothetic nominal power distribution. They are selected randomly in the following way:

One third of the cases are defined by $\lambda_i = 1$, $(i = 1, \ldots, E)$, i.e. no distortion with respect to the nominal distribution;

The rest of the cases are defined by $\lambda_j = 2$, for a randomly selected value of j, and $\lambda_i = 1$, $(i = 1, \ldots, j-1, j+1, \ldots, E)$, i.e. a higher participation of the jth element.

4.2.2. *Transient stability (pre)analysis and attribute calculation.* To use the data base for investigating the DTTS approach, we carried out the following preliminary calculations:

1. Approximate calculation of the CCT of the 2000 OPs, for a 3ϕSC at each one of the 31 generator buses, using the extended equal area criterion (EEAC) (Xue *et al.*, 1988). This gave us good information about the relative severity of these contingencies in relation to the OPs represented in the data base, and allowed us to select three "interesting" ones for our investigations.
2. Precise calculation of the CCTs of the OPs, for the three selected faults, using the step by step (SBS) method. To accelerate the iterative cut and try process delimiting the precise value of the CCT, the approximate values supplied by the EEAC were used as an initial guess. Incidentally, these latter were found to be in a very good agreement with those computed by the SBS method (over the 6000 CCT values we found a negative bias of −0.007s and a standard deviation of 0.032s of the CCTs provided by the EEAC as compared to those computed by the SBS method).
3. Generation of the files containing the attribute values for the 2000 OPs. About 270 different "primary" attributes have been computed, comprising zonal statistics on loads, generation and voltage, voltage magnitudes at all buses, voltage angles at important buses, active and reactive power of each generator, and topology information for each OP. Other attributes, such as line power flows, could be defined as simple algebraic combinations of the primary attributes and did not necessitate to be stored explicitly. The attribute files were constructed on the basis of the load flow data and state vectors.

Overall, the data base contains about 300 values per operating point, in addition to the input files required for the load flow and transient stability analysis programs.

4.3. *Simulation results*

Over 400 different DTs were built for various scenarios: three different contingencies, about 15 different classifications, distributed almost equally among the two-, three-, and four-class patterns, learning set sizes ranging from 100 to 1500 OPs, more than 100 different candidate attributes, α values ranging from 1.0 to 0.00005. The DTs were evaluated on the basis of independent test sets.

The resulting observations are organized and presented in six parts (Sections 4.3.1–4.3.6), although they are interdependent in many respects. The first part analyses general DT features, such as complexity and accuracy, with respect to various classification patterns, while fixing the other parameters (size of the LS, value of α, list of candidate attributes); this provides a good insight into the overall DTTS method. The second part focuses on the influence of α on the resulting DTs, and suggests how to select good α values in practical situations. The third part explores the way the size of the LS affects the DTs, while the fourth part discusses the influence of the candidate attributes on the DTs. The fifth part examines the influence of the number of classes on the misclassification rate and severity. Finally, the sixth part compares the different accuracy estimates defined in Section 2.8.

TABLE 2. TREE FEATURES AS RELATED TO THE NUMBER OF CLASSES. $\alpha = 0.0001$, $N = 500$, $M = 1500$

Gen.	Two classes			Three classes			Four classes		
bus #	N_t	$R^{ts}(\%)$	A	N_t	$R^{ts}(\%)$	A	N_t	$R^{ts}(\%)$	A
2	7	2.27	3	17	5.67	4	27	9.60	7
21	9	3.73	3	17	3.60	4	25	8.40	6
49	5	1.53	1	9	4.33	2	15	7.53	3

4.3.1. *General tree characteristics.* DTs corresponding to three different classifications patterns (of two, three and four classes) are built for each of the three different contingencies under the following conditions: the data base (2000 OPs) is divided in a LS composed of $N = 500$ OPs and a test set composed of the $M = 1500$ remaining OPs (this fairly large size provides a high precision to the R^{ts} estimate); the value of α is fixed to 0.0001 (the justification of this choice will be found below); and the list of candidate attributes, the same for each DT, is composed of 81 static variables.

The results are summarized in Table 2. The first column specifies the faulted bus number of the contingency. The nine following columns specify for the indicated contingency and number of classes, the characteristics of the resulting DT: N_t, the total number of its nodes (which measures its complexity); R^{ts}, its test sample estimate, representing the percentage of misclassified test states; A, the number of different retained attributes among the 81 candidates. This is repeated for the three contingencies, providing the features of nine DTs listed in the table.

The same set of investigations was repeated three more times, with three other learning sets of the same size (each of 500 states), in order to detect the variability of the DTs with the LS. The obtained results are very similar to the above, and induce the following conclusions:

—The complexity increases from the very simple two-class trees to the moderately complicated three- and four-class ones;
—Their accuracy is quite satisfactory especially in the two- and three-class cases; the four-class trees are less accurate but, as we discuss below, their errors are less harmful;
—The error rate varies only moderately with the fault location;
—The total number of retained attributes remains overall very small.

This latter aspect justifies the conjecture that transient stability is a localized phenomenon and highlights the ability of the method to select the relevant attributes. (A worth mentioning fact which does not appear in the table, is that the selected attributes are essentially variables related to buses very close to the faulted one.)

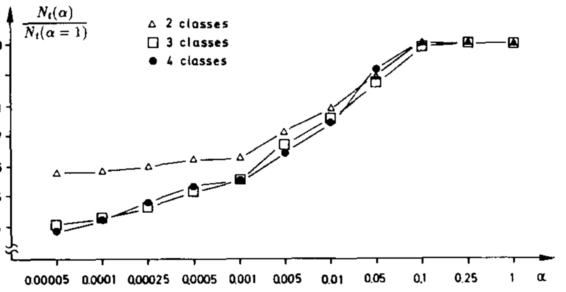

FIG. 5. Influence of α on the normalized number of nodes. $N = 500$, $M = 1500$.

4.3.2. *Impact of the parameter α.* For a given LS, attributes list and classification pattern, the principle of the stop splitting criterion suggests that the lower the value of α the smaller the resulting DT. Although this reduction in complexity is certainly interesting from many viewpoints, it could also cause the DTs to be less accurate, as indicated by our earlier discussions. Hence, the necessity of scrutinizing the effect of α on the complexity and accuracy of the DTs.

These observations were extensively investigated using different faults, numbers of classes, and learning sets. They yielded the following conclusions:

- The most important reduction of a DT's complexity is obtained as soon as α enters the range 0.001 to 0.0001; quantitatively, this effect is more marked in the three- and four-class cases;
- Although lower values of α still reduce (sometimes significantly) the size of the DTs, the effect is generally less spectacular;
- The DTs corresponding to $\alpha = 0.00005$ can be two to ten times smaller than those corresponding to $\alpha = 1.0$; besides:
- When α decreases from 0.005 to 0.00005, the accuracy of the DTs varies very slowly, and generally insignificantly (cf. the statistical uncertainty of R^{ts}).

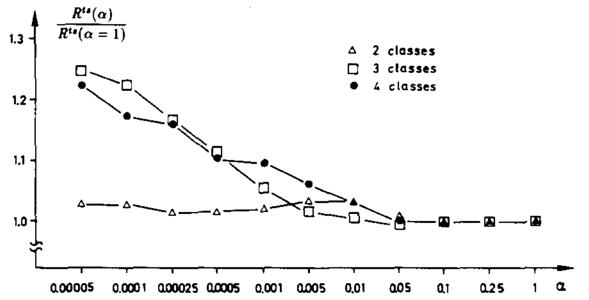

FIG. 6. Influence of α on the normalized test set error estimate. $N = 500$, $M = 1500$.

Figures 5 and 6 illustrate graphically the above observations in the two-, three-, and four-class situations. In each class, the curves represent the mean relative variations of the size, (N_t), and of the misclassification rate (R^{ts}), of 132 DTs (each point of the curves represents the mean value of 12 different DTs built for the corresponding value of α).

The following example provides an insight into the way the value of α operates in the particular case where the attributes contain less information (i.e. the "less separable" case). This corresponds to a LS of 500 states, a biclass pattern, and seven values of α ranging from 1.0 (the nodes are split until they are all leaves) to 0.00005 (extremely "cautious" behavior). Seven DTs were accordingly built, for the contingency #2, where the most important attribute (PG122, the active power generated at the bus #112) was removed from the list of candidate attributes. Table 3 summarizes the results. Columns 2–5 list the number of respectively test nodes, leaves, deadends, total. (Note that since the DTs are binary, they always have as many nodes as the double of test nodes plus one: $N_t = 2N_{ts} + 1$). The following three columns of Table 3 provide the misclassification rate as appraised by respectively the resubstitution, the test sample, and the cross-validation estimate; incidentally, observe the optimistic character of R^{ls}, and to a much lesser extent, of R^{cv}.

One can see that decreasing the value of α in the interval [1.0 . . . 0.01] not only drastically reduces the complexity of the DTs but it moreover significantly improves their accuracy. On the other hand, in the interval [0.01 . . . 0.00005] the size of the DTs decreases moderately, whereas their accuracy remains unchanged. Figure 7 is an eloquent illustration of the decrease in complexity: for $\alpha = 1$, N_t amounts to 63, whereas for $\alpha = 0.0001$ N_t reduces to 5. Notice that the two DTs have the same structure nearby their respective roots.

The general conclusions are the following:

1. The statistical hypothesis test is able to detect and identify the deadends in a very efficient and reliable manner, provided that the value of α is lower than 0.001;
2. Using α values below 0.001 provides the twofold benefit of reduced complexity and improved reliability; this effect is even more important in the less separable cases, where the "variance" effect can be very important;
3. The precise value of α, realizing the best compromise between what are called "variance" and "bias" in Section 2, lies somewhere in between 0.001 and 0.00005; the lower the number of classes, the lower the "optimal" value; moreover, the higher the contribution of the variance effect (e.g. the lesser the information contained in the candidate attributes), the lower the optimal value of α;
4. Anyhow, the bias effect remains very low (it would probably appear markedly for values of α much lower than 0.00005); thus the precise value of α is practically of no concern, as long as it lies in the range [0.001 . . . 0.00005];
5. Hence, considering that for a required accuracy, the smaller the trees the better, it is advisable to use $\alpha = 0.0001$ in all cases; sometimes it is even preferable to sacrifice a little accuracy for simplicity of the tree structure.

TABLE 3. IMPACT OF α ON COMPLEXITY AND ACCURACY IN THE LESS SEPARABLE CASE OF A BICLASS PATTERN $N = 500$, $M = 1500$, CONTINGENCY #2

α	N_{ts}	N_{lv}	N_{dd}	N_t	$R^{ls}(\%)$	$R^{ts}(\%)$	$R^{cv}(\%)$
1.00000	31	32	0	63	0.6	11.9	10.2
0.10000	27	26	2	55	1.0	11.5	7.4
0.01000	3	3	1	7	6.0	8.8	7.4
0.00500	3	3	1	7	6.0	8.8	7.4
0.00050	3	3	1	7	6.0	8.8	7.4
0.00010	2	2	1	5	6.0	8.8	7.2
0.00005	2	2	1	5	6.0	8.8	7.2

Remark. Although the above conclusions are drawn in the specific context of transient stability, preliminary investigations indicate that they correspond to the very nature of the statistical hypothesis test, and should remain valid in general. (Wehenkel, 1990).

4.3.3. *On the size of the LS.* One may distinguish the following three questions:

How does the size of the LS influence a DT's complexity and accuracy?

What is the minimal number of learning states required to achieve an acceptable accuracy?

How "stable" is the DT when the LS changes? or stated otherwise, how sensitive is the structure of the DT to the changes of the LS?

Qualitatively, according to the principle of the inductive inference method, one can say that, for a given value of α, increasing the size of the LS will generally (although not necessarily) increase the complexity and the accuracy.

Quantitatively, the answer to the first two questions strongly depends on the particular application of concern and especially on the intricacy of the underlying relationship between the attributes and the classification pattern. The more intricate this relation, the larger the sufficiently accurate DTs and the larger the LS required to build them in a reliable fashion.

As regards stability, we generally found that the nodes near the root of the DT (which

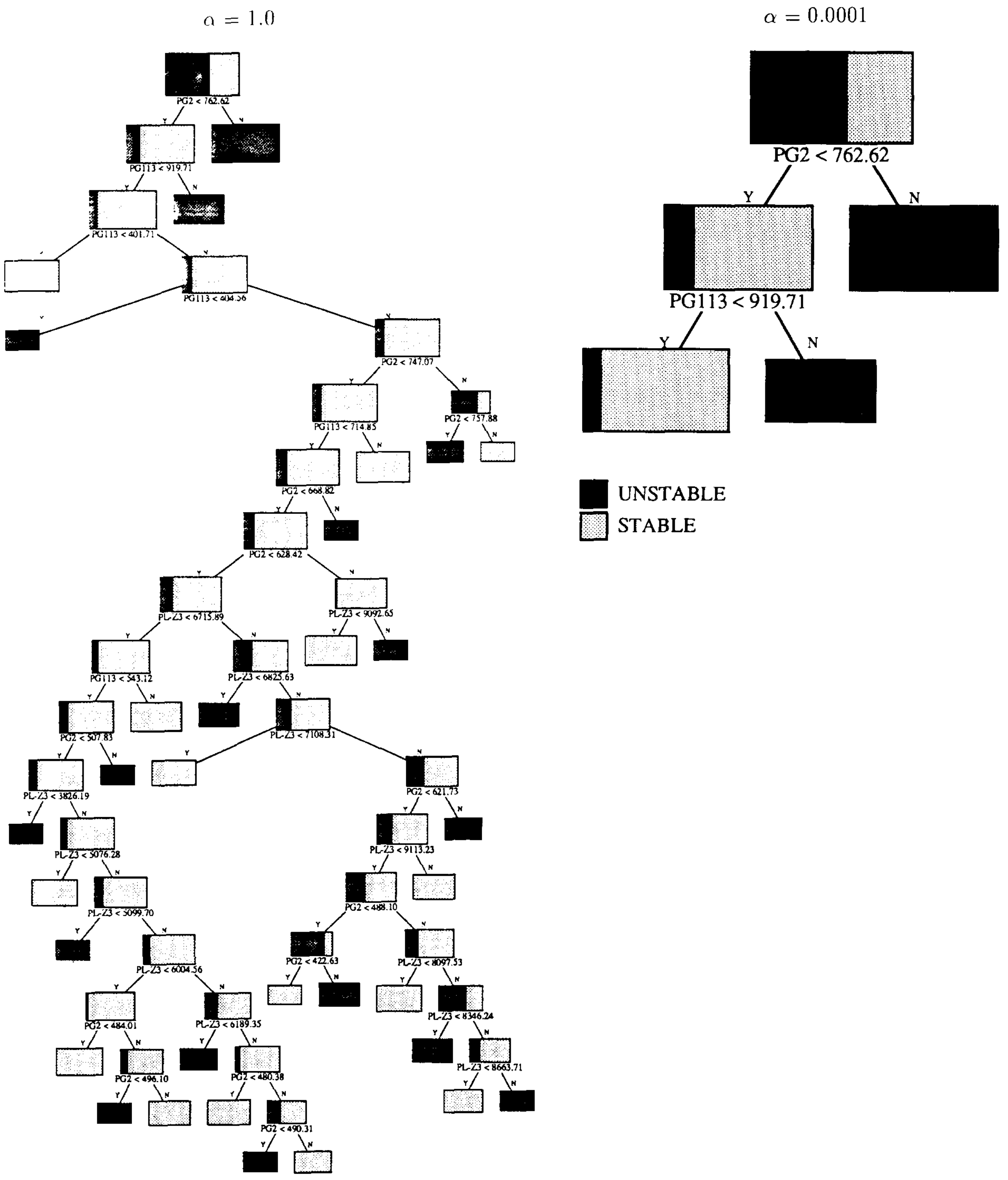

FIG. 7. Typical variation of tree structure with α (biclass DTs for contingency #2, $N = 500$).

express the most relevant relationships) do not change much when N increases, whereas the lower nodes (expressing the details) can change more significantly.

To illustrate and support these considerations, we conducted the following investigations: for the contingency #21, $\alpha = 0.00005$, and in the four-class pattern, six DTs were built with N, the size of the learning set, varying from 100 to 1250 states. Each DT was tested on the basis of the remaining $M = 2000 - N$ test states. The results are represented graphically in Figs. 8 and 9. One can see that for increasing values of N the error rate decreases from 12.2% to 5.5%, whereas the number of the trees' nodes increases from 9 to 43. At the same time, the number of *retained* attributes is found to increase from 2 to 11. Figure 10 provides two DTs, built for $N = 250$ and 750, and having respectively $N_t = 9$ and 25, and $A = 2$ and 7. Observe the stability of the DTs.

These and many other similar results indicate

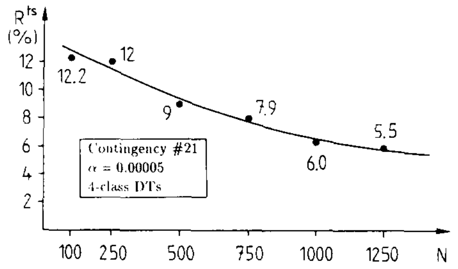

FIG. 8. Influence of the LS size, N, on the test set error estimate.

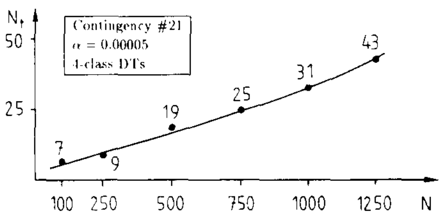

FIG. 9. Influence of the LS size, N, on the number of nodes.

that a size of 500 learning states seems to be a good compromise, allowing the building of sufficiently reliable and moderately complex DTs.

4.3.4. *Candidate attributes.* What would happen if the most relevant attribute (i.e. the one selected for the test at the tree root) were removed from the list of candidate attributes? and what if additional candidate attributes were provided to the procedure?

Removing from the candidate list the most significant attribute, causes a decrease in the accuracy of the DT; however, if "good" alternative attributes remain in the list, this degradation is rather restricted. A ground of comparison has been given by Tables 2 and 3. In Table 2 a DT composed of 7 nodes was obtained for the two-class case corresponding to contingency #2, when a sufficiently complete list of candidate attributes is used (including in particular the most relevant attribute for this case, namely PG_{112}, the active power generated at bus #112). Table 3 indicates the effect of removing PG_{112} from the list of candidate attributes: the tree corresponding to $\alpha = 0.0001$ reduces to 5 nodes and its probability of misclassification R^{ts} increases from 2.27% to 8.80%. This is further illustrated in Fig. 11, where the two biclass trees are presented: obviously, removing the best attribute, has caused a degradation of the tree's accuracy which, nevertheless, remains quite good (8.80% vs 2.27%).

Conversely, providing additional relevant attributes will generally improve the DT accuracy; however, if the most significant ones are already included in the list, only the lower parts of the DT will be affected, and the increase in accuracy will be almost negligible.

As a conclusion, if the important attributes are not known *a priori*, it is advisable to use an as large as possible list of candidate attributes. The first constructed DT identifies itself the relevant attributes, which can be used for subsequent tree constructions, possibly in addition to new ones.

4.3.5. *Misclassification errors as related to the number of classes.* In Section 4.3.1 we observed that the larger the number of a tree classes, and the larger its complexity and misclassification error rate. This is not surprising though: for a given LS, the more detailed the information one wants to extract, the larger its inaccuracy; as a counterpart, one may reasonably expect this inaccuracy to be less harmful.

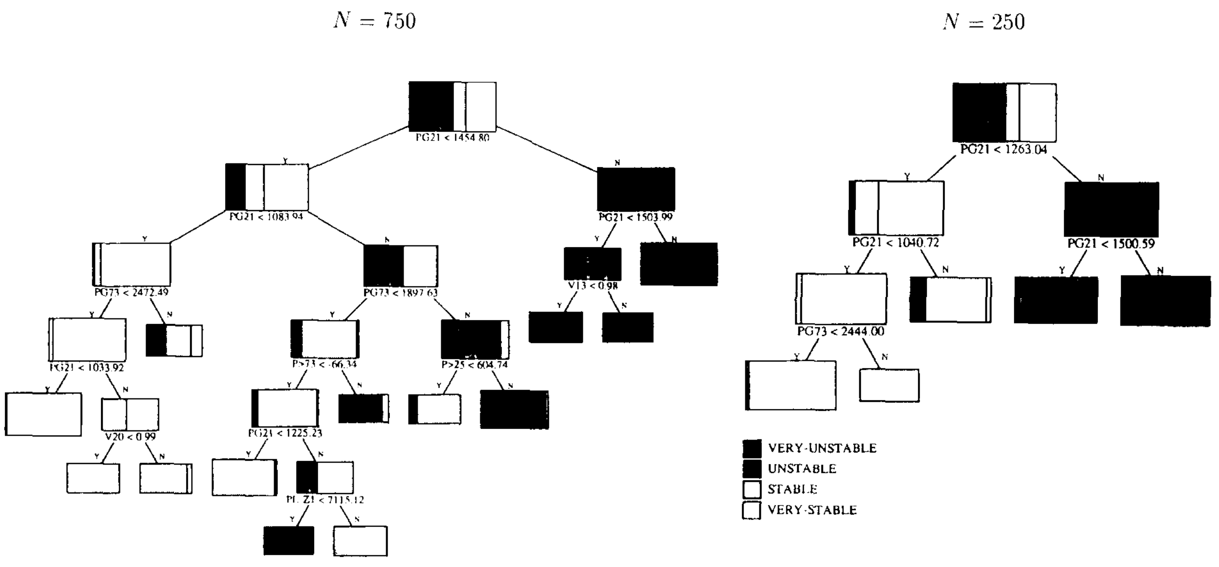

FIG. 10. Influence of the LS size, N, on the tree structure (contingency #21, $\alpha = 0.00005$).

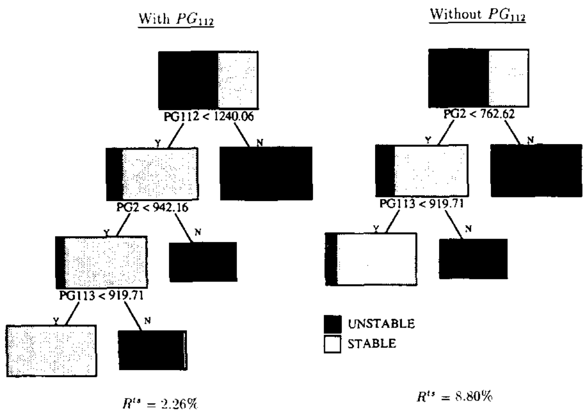

FIG. 11. Effect of removing the best attribute PG_{112} (contingency #2, $\alpha = 0.00005$, $N = 500$).

All our investigations corroborate this reasoning. A typical case is considered in Tables 4 and 5, which collect data for respectively a two- and a three-class tree, built for the contingency #21, $\alpha = 0.00005$, $N = 500$ and $M = 1500$. The true classification of these 1500 test states is reported in the columns of the tables, where the labels VU, U, S, VS stand for respectively very unstable, unstable, stable, very stable. Their classification as provided by the (biclass and four-class) trees, is reported in the rows of the tables; an additional row provides the misclassification rates. These latter are overall quite

TABLE 4. TYPICAL MISCLASSIFICATION ERRORS OF A TWO-CLASS TREE. $\alpha = 0.00005$, $N = 500$, $M = 1500$

		True classification		
		U	S	All
Tree classification	U	723	25	748
	S	56	696	752
	All	779	721	1500
Errors (%)		3.73	1.67	5.4

TABLE 5. TYPICAL MISCLASSIFICATION ERROR DISTRIBUTION OF A FOUR-CLASS TREE. $\alpha = 0.00005$, $N = 500$, $M = 1500$

		True classification				
		VU	U	S	VS	All
Tree classification	VU	461	29	0	0	490
	U	8	217	8	0	233
	S	7	56	189	8	260
	VS	0	1	20	496	517
	All	476	303	217	504	1500
Errors (%)		1.0	5.73	1.87	0.53	9.13

reasonable, as is also suggested by the strongly diagonal dominant character of the "kernel" of the tables.

Observe also that the total error of the biclass tree is lower than of the four-class tree; but the latter error is less misleading than the former; for example, declaring unstable a state which is actually stable is less misleading in the four-class than in the biclass tree, because of the finer definition of the four classes. Stated otherwise, in the four-class tree, a large majority of errors appear among neighboring classes (e.g. see Fig. 12): the error distribution concentrates mainly around the true class and the number of "outliers" almost (if not totally) reduces to zero. Observe also that a finer exploration and identification of the misclassified states indicates that in the biclass tree these latter mainly concentrate in the vicinity of the stable–unstable borderline as well; hence their misclassification is not totally misleading, after all.

4.3.6. *On the accuracy of the R^{ls} and R^{cv} estimates.* To quantify the considerations of Section 2.8, we have compared the accuracy of the R^{ls} and R^{cv} estimates with respect to R^{ts} considered as the benchmark, because of its high reliability.

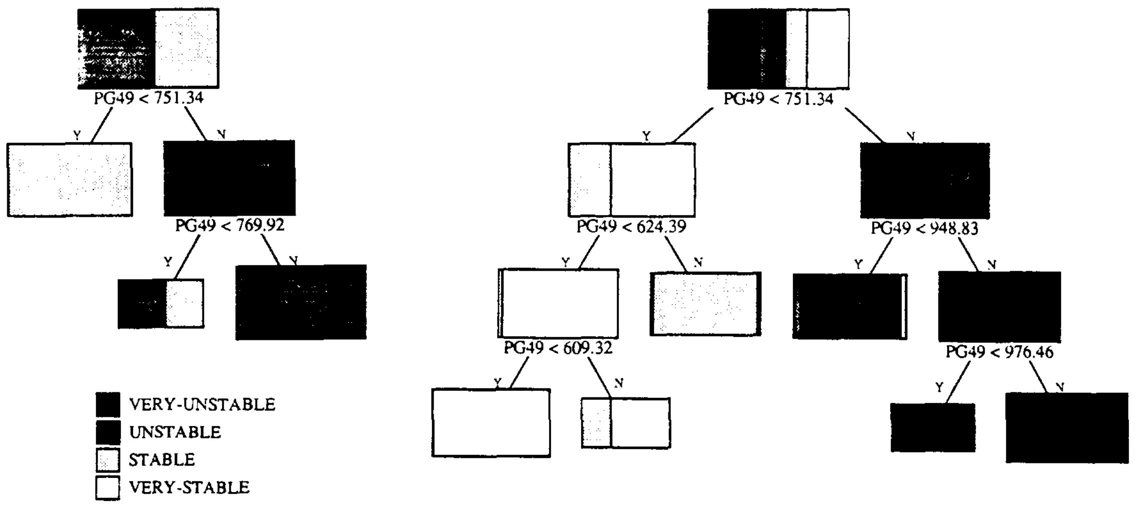

FIG. 12. Typical bi- and four-class trees (contingency #49, $\alpha = 0.00005$, $N = 500$).

With respect to R^{ls}, we observe the following:

- In all cases, it is strongly optimistically biased;
- The bias varies from almost 100% underestimation for the high values of α to approximatively 40% underestimation for $\alpha = 0.00005$;
- For a fixed value of α, the bias decreases slowly when the number N of learning states increases;
- The bias could become acceptable only for values of α much lower than 0.00005;
- Thus, for "good" values of α resulting from our earlier discussion, this measure is of no practical value for the estimation and comparison of accuracies.

Regarding the accuracy of R^{cv}, we observe the following:

- For a given DT, its precise value is almost independent of the value of the parameter V, as long as this value is larger than 10.
- R^{cv} sometimes overestimates, sometimes underestimates the DTs actual accuracy, depending on the particular DT of concern;
- Defining the error of R^{cv} by $E^{cv} \triangleq R^{cv} - R^{ts}$ we found that the mean value of E^{cv} over 14 different DTs and for different values of V, is -0.17%, whereas the standard deviation of E^{cv} is 2.15% (for the purpose of comparison we mention that the variance of R^{ts}, is about 0.7% for $M = 1500$, and about 2% for $M = 150$);
- This corroborates our earlier statement that the variance of R^{cv} is rather high;
- As a conclusion, this estimate is particularly interesting when the total number of observations $(N + M)$ is too small in practice to be split into an LS and a sufficiently large $(M > 500)$ test set.

4.4. *A global set of simulations*

Another set of simulations concerning a less refined but more global investigation than that of the previous Section 4.3 has been carried out using the same data base of 2000 OPs: 31 contingencies of the 3ϕSC type, successively applied at each generator bus of the 31-machine system which have been pre-analysed via the extended equal area criterion. Table 6 collects information for the two-, three-, and four-class patterns (N_t and A stand for the total number of nodes and of selected attributes). Its last line summarises the mean characteristics of the DTs in terms of the number of stability classes.

Note that their complexity and accuracy depend almost linearly on the number of classes. Observe also that in terms of accuracy, this global assessment is certainly pessimistic, unfavorable to the method. This is due to the fact

TABLE 6. GLOBAL ASSESSMENT 3ϕSC APPLIED AT EACH GENERATOR BUS $\alpha = 0.0001$, $N = 500$, $M = 1500$

Gen.	Two classes			Three classes			Four classes		
Bus #	N_t	$R^{ts}(\%)$	A	N_t	$R^{ts}(\%)$	A	N_t	$R^{ts}(\%)$	A
2	7	2.67	3	19	5.40	5	25	9.53	6
11	7	3.73	2	17	6.67	6	21	11.40	4
19	5	4.33	2	13	11.73	2	27	11.87	8
21	5	4.00	1	17	5.60	5	19	8.13	5
22	5	3.93	1	17	7.13	5	21	10.47	3
29	3	2.47	1	9	4.33	2	15	7.87	4
30	5	3.93	2	17	8.07	6	23	9.60	6
39	3	2.07	1	7	4.27	1	9	5.60	2
41	7	3.87	2	19	13.20	5	25	12.67	6
44	9	2.73	2	13	6.20	5	15	9.20	3
48	15	4.67	4	23	8.27	7	27	15.87	6
49	3	1.93	1	7	5.07	1	13	9.47	2
50	15	3.73	5	23	9.13	7	29	11.80	9
59	7	5.13	3	17	7.33	5	23	12.33	5
65	5	2.27	1	15	3.40	3	11	6.73	2
66	3	1.67	1	9	3.40	3	17	6.33	4
70	7	3.00	2	11	7.93	3	19	7.93	5
71	7	3.27	2	15	8.40	5	21	12.67	6
72	5	2.20	2	9	5.33	3	11	9.20	3
73	3	2.73	1	11	5.13	3	19	7.87	4
74	17	4.20	6	23	8.87	6	31	17.07	9
75	5	3.13	1	11	3.87	2	19	10.27	6
79	3	2.00	1	9	4.80	2	15	7.40	4
84	9	4.33	3	13	8.20	3	25	10.40	7
95	5	2.00	2	15	6.13	4	21	8.67	4
112	9	1.93	3	15	5.00	3	15	5.40	3
113	5	5.00	2	15	8.67	5	29	11.40	8
123	11	5.60	4	9	8.13	3	31	9.73	9
124	7	2.53	2	15	3.80	4	15	9.47	3
129	7	4.00	2	19	9.47	5	17	13.53	6
132	7	1.73	2	7	5.80	1	13	8.47	3
Mean values over 31 contingencies									
	6.8	3.25	2.2	14.2	6.72	3.9	20	9.95	5.0

that the candidate attributes used to construct the trees have been chosen so as to cover the whole power system; no particular care has been taken to consider a more refined list of attributes around the contingencies locations. (Remember that for the 31 contingencies, only 83 attributes are used, often not close enough to some of them.) Another, although less important source of inaccuracy is the fact that the CCTs used for classifying the OPs of both the LS and the TS have been computed via the EEAC and not the numerical integration method; this, unavoidably introduces a small bias in most cases. More detailed information may be found in Wehenkel (1990).

5. DISCUSSION

The questions posed and the answers given in the preceding sections are reorganized so as to draw general conclusions. Some pertain specifically to the DTTS method, some others apply to the DT methodology in general.

The statistical test used to stop splitting the nodes of a grown tree appears to work very satisfactorily. Its threshold α provides an effective tool for controlling the complexity and the accuracy of the resulting tree. It is particularly interesting that simplicity and accuracy of a DT are not contradictory objectives, at least in a rather large range of the α values. Indeed, in this range, decreasing α does not significantly affect the accuracy, whereas it contributes to drastically decrease the number of nodes of the tree. Incidentally, this explains *a posteriori* the good performances obtained in our earlier investigations where α was fixed rather arbitrarily to 0.01. Overall, the simplicity and accuracy of the trees provided by the method are quite remarkable. This seems to be a general feature of the inductive inference method we developed to build DTs.

The particular DTTS approach proves also to be very effective in many respects, and the underlying conjectures fully justified. For example, among the large number of attributes proposed to the method, only a few are retained as the relevant parameters driving the transient stability phenomena. Moreover, increasing the number of classification patterns of a tree provides additional relevant attributes; this allows to get a more refined insight into the mechanism of the phenomena, and to offer additional means to control transient stability. The interplay between biclass trees—with extremely simple structures and reduced number of relevant attributes, and multiclass trees—with more complex (yet tractable) structures and larger (yet restricted) number of attributes, is another attractive feature of the method, which thus shows to be very flexible and stable. This stability of a tree with respect to the attributes is a very interesting aspect, indeed: it amounts to systematically using the same, more relevant attributes nearby its root, whatever the number of its classes and its complexity.

The tradeoff between bi- and multiclass DTs appears thus to be a great asset of the DTTS method, not a drawback. Various solutions, supplementing different, complementary information extracted from a LS, may thus be exploited for various purposes, even at the price of a somewhat lesser accuracy in the multiclass case.

Investigations relating to the construction of an "adequate" LS have pointed out another interesting aspect, namely that reasonable sizes of LSs are sufficient to build reliable DTs. On the other hand, the states composing the LSs were chosen on a statistical basis, the purpose sought here being the objective assessment of the DTTS method. Note that in a real world context, this choice should take into account requirements imposed by the power system of concern, specified in collaboration with the engineers and operators in charge of the system. Reconsidering the "right size" of a LS, one may wonder to which extent this should depend on the size of the power system. To answer this question, one probably should specify whether the purpose is to build trees for contingencies spread throughout the whole system, or whether one seeks to explore some particular contingencies. In the former case, a LS covering the whole power system, with sufficiently detailed information, would indeed be needed; its size would therefore increase with that of the power system. In the latter case, the LS should essentially contain detailed information only for the regions of concern for these contingencies.

One could object that the above conclusions rely on the particular, very severe type of contingencies considered sofar, and also on particularly simplified system modelling. However, our objective was the validation of the method as such; we believe that this objective has been encountered.

Certainly, to be interesting a method has to be computationally tractable. The considerations of Section 3.1.4 indicate that the main, and in fact sole, really burdensome task is the construction of the data base. But this has to be done only once, then occasionally updated. The construction of the DTs, although also off-line, is a less demanding task; and the better the knowledge of the power system, the faster the construction of its trees. Once constructed, the storage and use of the DTs are extremely inexpensive; thousands of DTs could be simultaneously stored into main memory; as for the mean classification time, it was assessed to be about 0.6 ms, i.e. almost negligible.

The way of using the DTs was not considered in this paper; only classes of possible uses were enumerated, and prospects for their real-time applications to transient stability analysis, sensitivity analysis and preventive control were suggested. Nevertheless, other interesting applications of the DTTS approach may be foreseen as well, in the context of training simulators, and of planning studies.

6. CONCLUSION

Two main objectives have been pursued in this paper. First, to get in-depth knowledge of the inductive inference method designed in our previous studies, and more specifically of its stop splitting criterion. Second, to scrutinize the basic features of the decision tree transient stability (DTTS) method, i.e. of the inductive inference method as applied to transient stability of power systems.

To encounter these objectives, a large-scale investigation was conducted, using a realistic power system. The obtained results are quite interesting. As regards the inductive inference method, it was proven to be very efficient, indeed, appropriate to yield simple and accurate DTs; and although it would be hazardous to compare methods used in different application domains, one nevertheless may say that it appears to be among the very effective methods reported in the technical literature.

Concerning its application to transient stability, the devised DTTS method has exhibited very attractive features, with manifold potential. For example, it was found to be capable of treating the three aspects of transient stability assessment, viz. analysis, sensitivity, and control. It could therefore be exploited in planning studies. At least as interesting are its real-time aspects, and its capabilities in on-line transient stability assessment and preventive control.

Admittedly, many other questions still remain unexplored. This study was the first, indispensable step towards real world application of the DTTS method.

REFERENCES

Bergen, A. R. (1986). *Power System Analysis*. Prentice Hall, Englewood Cliffs, NJ.

Breiman, L., J. H. Friedman, R. A. Olshen and C. J. Stone (1984). *Classification and Regression Trees*. Belmont, Wadsworth.

EPRI EL-4958 Project 2496-1 (1987). Dynamic Security Assessment for Power Systems: Research Plan. Final Report.

Friedman, J. H. (1977). A recursive partitioning decision rule for nonparametric classification. *IEEE Trans. Computers*, **C-26**, 404–408.

Kononenko, I., I. Bratko and E. Roskar (1984). Experiments in automatic learning of medical diagnosis rules. Technical Report. Jozef Stefan Institute, Ljubljana, Yugoslavia.

Kvålseth, T. O. (1987). Entropy and correlation: some comments. *IEEE Trans. Syst, Man Cybern.*, **SMC-17,** 517–519.

Lee, S. T. Y. (1972). *Transient stability equivalents for power system planning*. Ph.D. Thesis, Massachusetts Institute of Technology, MA.

Quinlan, J. R. (1984). Learning efficient classification procedures and their application to chess endgames. In R. S. Michalski, J. G. Carbonell and T. M. Mitchel. *Machine Learning: An Artificial Intelligence Approach,* pp. 463–482. Springer, Berlin.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning,* **1,** 81–106.

Ribbens-Pavella, M., and F. J. Evans (1985). Direct methods for studying dynamics of large scale electric power systems—A survey. *Automatica,* **21, 1,** 1–21.

Toussaint, G. T. (1974). Bibliography on estimation of misclassification. *IEEE Trans. Inf. Theory,* **IT-120,** 472–479.

Wehenkel, L., Th. Van Cutsem and M. Ribbens-Pavella (1986). Artificial intelligence applied to on-line transient stability assessment of electric power systems. *Proc. 25th IEEE Conf. Decision and Control,* pp. 649–650. Athens, Greece.

Wehenkel, L., Th. Van Cutsem and M. Ribbens-Pavella (1987). Artificial intelligence applied to on-line transient stability assessment of electric power systems. *Proc. 10th IFAC World Congress,* pp. 308–313, Munich, F.R.C.

Wehenkel, L., Th. Van Cutsem and M. Ribbens-Pavella (1988). Decision trees applied to on-line transient stability assessment of power systems. *Proc. IEEE Int. Symp. on Circuits and Systems,* Vol. 2, pp. 1887–1890, Helsinki, Finland.

Wehenkel, L. (1988). Artificial intelligence methods for on-line transient stability assessment of electric power systems. *Proc. Symp. on Expert Systems Application to Power Systems,* pp. 5.1–5.8, Stockholm–Helsinki.

Wehenkel, L., Th. Van Cutsem and M. Ribbens-Pavella (1989a). Inductive inference applied to on-line transient stability assessment of electric power systems. *Automatica,* **25,** 445–451.

Wehenkel, L., Th. Van Cutsem and M. Ribbens-Pavella (1989b). An artificial intelligence framework for on-line transient stability assessment of power systems. *IEEE Transactions Power Systems,* **PWRS-4,** 789–800.

Wehenkel, L. (1990). *Une Approche de l'Intelligence Artificielle Appliquée à l'Evaluation de la Stabilité Transitoire des Réseaux Electriques,* Ph.D. Thesis (in French), University of Liège, Belgium.

Xue, Y., Th. Van Cutsem and M. Ribbens-Pavella (1988). A simple direct method for fast transient stability assessment of large power systems. *IEEE Trans. Power Systems,* **PWRS-3,** 400–421.