Dynamic Security Border Identification Using Enhanced Particle Swarm Optimization

Ioannis N. Kassabalidis, *Student Member, IEEE*, Mohamed A. El-Sharkawi, *Fellow, IEEE*, Robert J. Marks, II, *Fellow, IEEE*, Luciano S. Moulin, *Student Member, IEEE*, and Alexandre P. Alves da Silva, *Senior Member, IEEE*

***Abstract*—The ongoing deregulation of the energy market increases the need to operate modern power systems close to the *security border*. This requires enhanced methods for the *vulnerability border tracking*. The high-dimensional nature of power systems' operating space makes this difficult. However, new multiagent search techniques such as *particle swarm optimization* have shown great promise in handling high-dimensional nonlinear problems. This paper investigates the use of a new variation of particle swarm optimization to identify points on the security border of the power system, thereby identifying a vulnerability margin metric for the operating point.**

***Index Terms*—Border tracking, particle swarm optimization, security assessment, system dynamics.**

I. Introduction

In the last few decades, dynamic security evaluation (DSE) methods have been developed based on time-domain simulations and direct methods of energy function evaluation [1]–[4]. Despite advances in computer processing time, these methods, although potentially identifying the security status of the system, still require a time-consuming computational effort to calculate accurate and reliable security indices or margins. This is due to the structural complexity and wide operating ranges of modern power systems.

With the current trend toward deregulation and the participation of diverse players in the power market, power systems are being required to operate closer than ever to their security border [5]. The market's economic constraints may force the system to move even nearer to its security limits or vulnerability borders thereby decreasing the overall system's security margins [6]. Under this scenario, the operator must be acutely aware of the location of the security border in the operating space. Providing this data is a highly computational process that cannot be achieved by a model-based approach. The problem is even more complex because the border is a dynamically changing surface.

In recent years different approaches and new technologies have been applied to automate the DSE process and cope with the new challenges while simultaneously maintaining accuracy.

Manuscript received May 22, 2001; revised February 25, 2002. This work supported in part by the National Science Foundation, EPRI, and the Department of Defense.

I. N. Kassabalidis, M. A. El-Sharkawi, and R. J. Marks, II are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195-2500 USA (e-mail: elsharkawi@ee.washington.edu).

L. S. Moulin and A. P. Alves da Silva are with the Institute of Electrical Engineering, Federal Engineering School at Itajuba, EFEI, Itajuba, Brazil.

Publisher Item Identifier 10.1109/TPWRS.2002.800942.

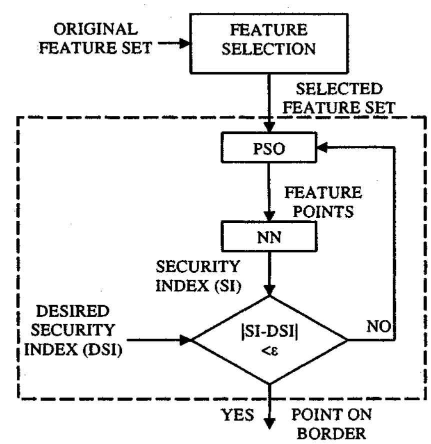

Fig. 1. Flow diagram for dynamic security border identification using enhanced particle swarm optimization.

Intelligent system technologies have provided promising applications in DSE. Examples include decision trees [7], [8], neural networks (NNs) [9]–[13], expert systems [14] and integrated techniques [15], [16]. Neural networks have received the most attention, where operating states are correlated with transient stability assessment [11], [12] or a security margin [9], [13].

Variations of this approach use the NN to visualize [17] or identify [18] the stability border. By the technique in [17], two-dimensional (2-D) visualizations of the security boundary can be obtained by varying only two independent critical parameters. In [18], offline simulations are used to train a NN in a query-based learning process to enhance the accuracy of the NN in the vicinity of a high order security border. Once trained, the NN is inverted to identify points on the border of the power system. This is a computationally intensive process.

We propose a technique where by the security border can be identified more quickly. Moreover, the border can be dynamically updated to reflect changes in system's topology due to faults or reconfigurations. The proposed technique has two steps. First, the main features for security border identification are selected. In the second step, an enhanced particle swarm optimization procedure is used as a fast constraint search technique to identify the border or the section of the border close to the current operating conditions.

The overall general system is shown in Fig. 1. Feature selection is performed both to overcome the NN curse of dimensionality, and to allow fast execution of particle swarm optimization

(PSO). The use of a NN as a security estimator is fundamental for a fast calculation of border operating points.

The procedure for border tracking is depicted in more detail inside the dashed box. PSO uses a NN as a predictor of security index values. The security index is compared with the desired security index. If the difference is small, the algorithm exits and produces the points on the border. Otherwise, the loop continues until it reaches a satisfactory solution or the maximum number of iterations is exceeded.

II. Feature Selection

For large-scale power systems, the dimensionality of the input space for security assessment is high [9], [10], [13]. This leads to computational problems for the boundary tracking algorithm, as well as the performance of the NN. The problems include the need for unrealistically large training data that spans the operating space, the likelihood that the NN memorizes the training data, the need for extensive training time, and the possibility of high testing errors when the system operation drifts [19]. Furthermore, a high dimensional input space increases the computational time needed for border identification. To decrease the dimensionality of the feature space while maintaining the classification accuracy, feature selection or extraction should be implemented [20].

Feature selection is applied to the data set before the NN is trained. The training set T is represented by ordered pairs $(\mathbf{x_i}, c_i)$, $T = \{\mathbf{x_i}, c_i\}_{i=1}^{N}$, where $\mathbf{x_i}$ is an n-*dimensional* vector representing the operating point and c_i is the security index of that point. The main objective of the feature selection technique is to generate a d-*dimensional* feature vector, $\mathbf{f}$, where $d < n$. The components of the feature vector are selected from the original n-dimensional input vector. The "d" selected features represent the original data in a new training set $T' = \{\mathbf{f_i}, c_i\}_{i=1}^{N}$. The training of the NN is considered successful if the mean and variance of the training and testing error of the NN using the d-dimensional vector is comparable to the mean and variance of the training and testing error of the NN using the n-dimensional original input vector.

One way to achieve feature selection is through exhaustive search by examining all possible subsets with respect to some evaluation function. The number of possibilities $\sum_{d=0}^{n}\binom{n}{d} = 2^n$ grows exponentially with the size of n, making exhaustive search impractical.

A more effective method based on genetic algorithms (GA) is proposed in [20]. In this method, a distance measure based on the Fisher's discriminant (FD) distance is used as the evaluation function [21].

An example of the feature selection performance is shown in Fig. 2. The figure is for the IEEE 50 generators system [22]. The original number of features is 102 representing all generated active and reactive powers, and the total system load's active and reactive power. The NN is trained to identify the system's critical clearing time (CCT), which is used as a security index. After the GA-based feature selection technique is applied, the original feature vector is reduced to a cardinality of only 10. The figure shows the testing error for both the n-dimensional and the d-dimensional NN's. The error is the difference between the actual target value and the NN output. As seen in the figures, the errors are comparable and the selected features should represent the security status of the system accurately. In the case when the accuracy must be higher, more features can be selected.

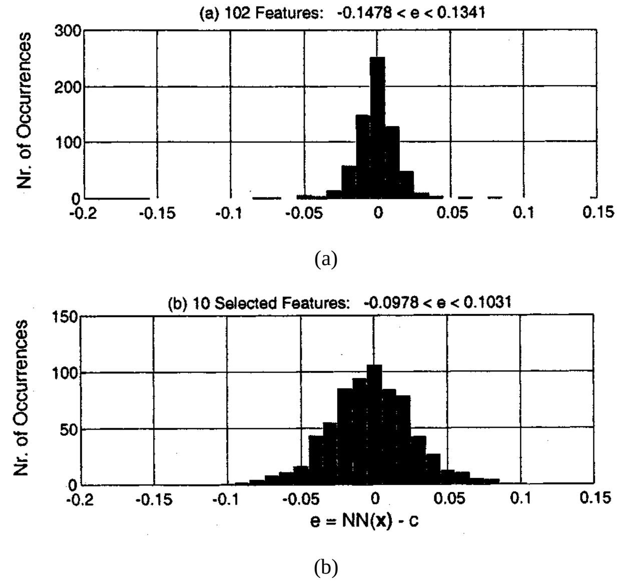

Fig. 2. NN performance on the test set with (a) 102 input features and (b) ten input features.

For this specific system, the advantages of feature selection are the following:

a) decrease of the training time of the NN;
b) increase of the speed of convergence of our PSO border identification algorithm.

The training time of the NN with ten features is 1.7 min and that of the 102-feature NN is 30 min, both on an old PII-233Mhz PC. For the speed of PSO convergence, it should be noted that a higher-dimensional feature space causes both slower convergence and decreased accuracy. In order to tackle this problem, more particles per swarm have to be used, thus increasing the computation time. The increase in particles would follow the *curse of dimensionality* rule [19], thus making PSO inefficient.

III. Particle Swarm Optimization

Due to the complexity of the power system, it is not possible to determine the dynamic security border analytically. An alternative is to use a search technique to place as many points on the border as needed to achieve desired interpolation accuracy [23]. This process requires the use of a fast dynamic security assessment technique and a rapid optimization search algorithm. The NN is used as a fast assessment tool and particle swarm optimization (PSO) is used as the search technique.

PSO is a novel optimization method developed by Eberhart, *et al.* [24]–[26]. It is a multiagent search technique that traces its evolution to the emergent motion of a flock of birds searching for food. It uses a number of agents (particles) that constitute a swarm. Each agent (particle) traverses the search space looking for the global minimum (or maximum). PSO has been recently proposed for power systems applications such as those reported in [27]. The goal of PSO is to find operating points that lie on the border.

While the agents in the PSO algorithm are searching the space, each agent remembers two positions. The first is the position of the best point the agent has found (self best). The second one is the position of the best point found among all agents (group best). The equations that govern the motion of each agent are

$$\begin{aligned}\vec{s}(k+1) &= \vec{s}(k) + \vec{v}(k)\\ \vec{v}(k+1) &= w.\vec{v}(k) + a_1.r(0,\,1).(\vec{s}_{selflbest}(k) - \vec{s}(k))\\ &\quad + a_2.r(0,\,1).(\vec{s}_{groupbest}(k) - \vec{s}(k)) \qquad (1)\end{aligned}$$

where $\vec{s}$ is a solution vector of a single particle, $\vec{v}$ is the velocity of this particle, a_1 and a_2 are two scalar parameters of the algorithm, w is an inertia weight, $r(0,\,1)$ is a uniform random number between 0 and 1, $grouplbest$ is the best solution of all particles and $selfbest$ is the best solution observed by the current particle. A maximum velocity, $v_{\max}$ that cannot be exceeded may also be imposed.

It has been found through experimentation [24]–[26] that the design parameter values that work best are $a_1 = a_2 = 2$, and w between 0.9 and 1.2.

IV. PSO Variation for Border Tracking

The goal of border tracking is to identify as many points as possible on the border. Therefore, we initiate a separate swarm for each point to be placed on the border. Each point is sequentially identified by running one swarm after the other. This method also provides the ability to impose constraints on the distribution of the points placed on the border. For this reason, we propose the following three different approaches to the border identification problem:

Case 1: Cover the entire security border with a predetermined number of points, without restrictions on the distribution of the points on the border.

Case 2: Place points uniformly on the entire security border.

Case 3: Uniformly cover the portion of the security border that is closest to the operating state.

In the first approach, our goal is to find points as accurately on the border as possible. The PSO variation requires an objective function to perform this task. In our dynamic security application, we seek to minimize the objective function

$$f_1(\vec{x}) = |NN(\vec{x}) - c| \qquad (2)$$

where NN is the security index produced by the neural network, $\vec{x}$ is the current position of the particle (operating point of the power system) and c is the security index of the border and is set by the user.

In the second approach, proximity to the closest neighbor is penalized, where neighbors are defined as the points found by previous swarms. The objective function in this case has a penalty term for the closeness on border point as follows:

$$f_2(\vec{x}) = f_1(\vec{x}) - w_c * d_c \qquad (3)$$

where $\vec{x}$ is the current position of the particle, d_c is the distance to the closest border point, and w_c is the weighting factor for d_c.

Covering the entire border with points in a high-dimensional power system, however, is very time-consuming. This is because the number of border points, as a rule of thumb, increases exponentially with the increase in the feature space dimensionality. The third approach addresses this issue. In order to accomplish this, we modify the fitness function to reward for proximity of the border point to the current operating state. In addition, the penalty for closeness of border points is maintained. These are two contradicting demands and must be balanced through appropriate weighting in the objective function

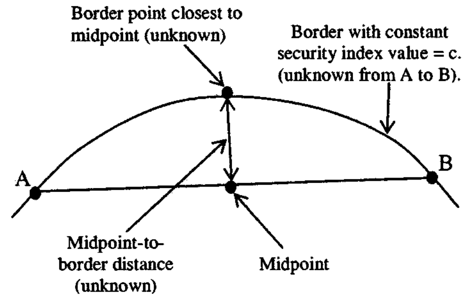

Fig. 3. Midpoint and its proximity to the border in the input space.

$$f_3(\vec{x}) = f_1(\vec{x}) - w_c * d_c + w_o * d_o \qquad (4)$$

where d_o is the distance from the operating point and w_o is the weighting factor for d_o.

These weighting factors affect the objective function in a very crucial manner and their choice should be such that the desired effect is accomplished. In our case, the desired effect is to be able to find points on the border, while imposing restrictions on their distance from each other, which is done through w_c, or their proximity to the current operating state, which is done through w_o. These restrictions can be conflicting with the goal of finding points close to the border, since points already on the border can, in effect, "push" the others away from the current operating state, while the current operating state can attract the points toward it and thus away from the border. Thus we choose w_c and w_o so that the maximum of the products $w_{c^*}\ d_c$ and $w_o * d_o$ does not exceed 1% of the of the desired border value. Also, if w_o is greater than w_c the points will tend to be close to the operating state but not as uniformly distributed, while if w_c is greater than w_o the points will tend to be more uniformly distributed but not as close to the current operating state.

To determine if enough points are on the border, we propose a technique utilizing the midpoints between neighboring border points. The midpoint is defined as the point derived by linear interpolation between a pair of closest neighbors. Each point on the border has one closest neighbor, thus the total number of midpoints is at least equal to half the number of points, but not greater than the total number of points. In Fig. 3, we show the midpoint of A and B. By definition, no other points exist on the segment AB. The proximity of the midpoint to the border can only be determined in the output space, since the segment AB is unknown. Thus, we choose to calculate the distance in the output space. We use this distance to determine whether enough points are placed on the border.

V. Two-Dimensional Border Identification Example

Before addressing the higher dimensional problem of power systems, we use a 2-D test case to illustrate the proposed al-

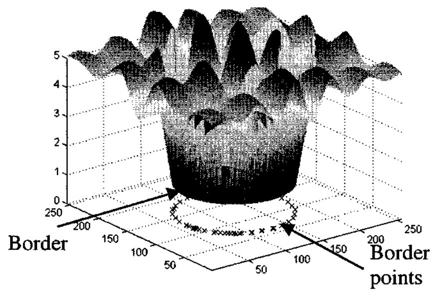

Fig. 4. Results for finding points about a distributed minimum. In this case, the objective function reaches a minimum around a circle. The points around this minima, using Case 2 of the enhanced particle swarm, are shown as $\times$s on the bottom plane.

gorithm. We use a 2-D surface that has a constant value for its ridge. The surface is used as an error function for the search algorithm. The purpose of the algorithm is to place as many points as possible on that ridge. The gradient of the surface around the ridge is not constant. The challenge in this case is to try to distribute the points uniformly on the border. Since the gradient on the border is not constant, more points tend to cluster in the high gradient region [18].

To address this problem, the three cases mentioned in Section IV are applied. In Fig. 4, we show the results for the second case only, where the points are spread uniformly on the border. As seen in Fig. 4, all points are nearly uniformly distributed on the ridge. For the first case, the points tend to gather around the area with the highest gradient, while, in the third case, they tend to gather around the current operating point in a uniformly distributed manner. In this simple example, we use a total of 32 swarms, with 10 particles, each initialized randomly. We set $a_1 = a_2 = 2$, $w = 1$ and $v_{\max} = 0.1$. For the parameters of the objective function, we set $w_c = 10^{-3}$ and $w_o = 2 \times 10^{-3}$.

VI. Practical Application of Enhanced PSO and Test System Description

We discuss here the issues concerning the practical application of our method. Our goal is to provide a fast and accurate tool for dynamic security border identification, which can be used by the system operator to avoid potentially vulnerable states. Fig. 5 shows the basic steps.

1) The dimensionality of the current operating state is reduced using feature selection.
2) Identify the border close to the operating state, in the selected feature space.
3) Use the distances in the feature-space as yardsticks for preventive control. Since the selected features are the most important of the whole feature set, based on the Fisher distance [20], [28], these distances represent the most important rules to prevent the system from getting close to the border.

This whole process relies on a NN trained for a specific fault and a specific system topology. In order to be able to handle a variety of faults and different topologies, it is necessary to have NNs that can cover these situations. Fortunately, it is possible to cover a range of similar topologies by a single NN, thereby decreasing the number of NNs that need to be trained [28]. However, extensive topological changes may require different NN's to be trained and a new boundary to be computed.

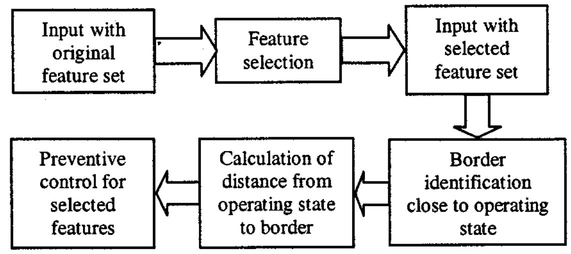

Fig. 5. Preventive control for feature selection.

This raises the question of which cases of the enhanced PSO can be used offline and which online. Cases 1 and 2 are both offline, since they place points on the entire border, which can be time-consuming. However, Case 3 can be used online, since only a small section of the border has to be identified. This means that a smaller number of points is necessary. As a matter of fact, PSO finds points in a sequential manner. Therefore, some information about the border is available once the first few points are located. This information is further refined once new points are discovered.

As a test system, we use the IEEE 50 generator system [22]. The inputs to the NN are ten features, selected by the GA technique mentioned in Section II. The system was modeled using ETMSP (Extended Transient-Midterm Stability Program) from the PSAPAC 5.0 software package supplied by EPRI. The vulnerability border is determined by the clearing time of the protection devices. The purpose of the NN is to predict the critical clearing time of a given operating point. We arbitrarily select the clearing time of the protection device at 0.45 s. We define the clearing time margin (CTM) as the difference between the CCT of a point and the clearing time of the protection devices.

For the PSO parameters, we use 20 particles in each swarm. Also, we set $a1 = a2 = 2$, $w = 1$ and $v_{\max} = 0.1$. These values were determined through experimentation. For the parameters of the fitness function, we set $w_c = 10^{-3}$ and $w_o = 2 * 10^{-3}$.

VII. Test Results

We implement all three cases of PSO, as described in Section IV, for the system described in Section VI. The results are presented in Fig. 6, Table I, and Fig. 7.

In Fig. 6, we present the histograms of the results for Case 1 with 5000 points placed on the border, Case 1 with 20 000 points, Case 2 with 5000 points and Case 3 with 5000 points. Each row in Fig. 6 corresponds to one case. In Fig. 6, Fig. 7 and Table I the labels a), b), c), or d) have the following meaning.

a) The distribution of CTM of the points. This identifies the distance of the points from the border in the output space.
b) The distribution of the CTM of the midpoints. This serves as an indicator of whether the points are enough to represent the border by interpolation, as explained in Section IV.

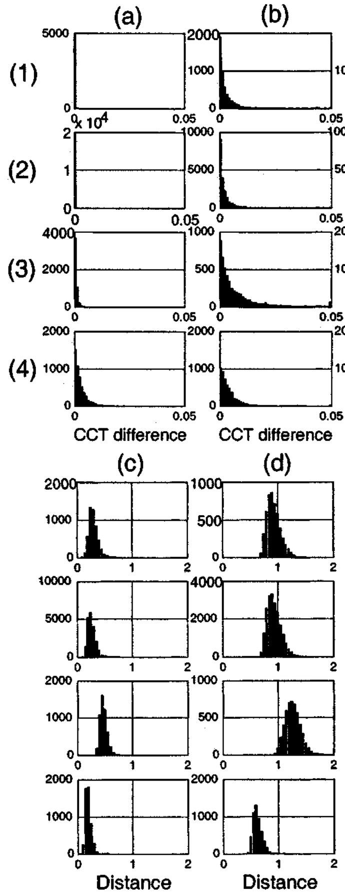

Fig. 6. Result histograms for all cases. (1) Case 1–5000 pts. (2) Case 1–20 000 pts. (3) Case 2–5000 pts. (4) Case 3–5000 pts.

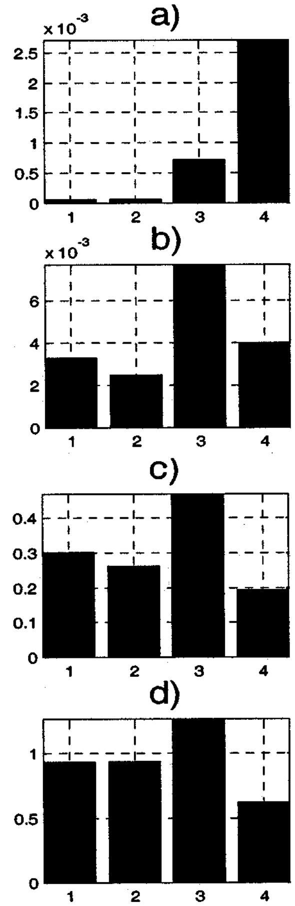

Fig. 7. Plots of means for a)–d) for all the cases.

c) The distribution of the distance of each point from its closest neighbor. This distribution is an indicator of the density and uniformity of border coverage.

d) The distribution of the average distance of each point from all the other points. This is an indicator of the size of the region of the border that is covered by the points.

Also, in Table I, μ and σ stand for mean and standard deviation respectively.

From Fig. 6 and Table I, we observe the following for Case 1 with 5000 points.

- Almost all the points are on the border, as shown in a).
- The midpoints are close to the border, as shown in b).

Hence, the portion of the border where most of the points are located is covered satisfactorily. From c) and d) we derive no conclusions yet, rather we expect differences with cases 2 and 3.

The next step is to investigate whether an increase in the number of points produces any significant changes in the results. Thus we increase the number of border points to 20 000. Again, from Fig. 6 and Table I we observe the following for Case 1 with 20 000 points.

- Almost all the points are on the border, as shown in a).
- The midpoints are close to the border, as shown in b).
- There is a decrease in the mean distance to the closest neighbor, shown in c), as compared to Case 1-5000.

TABLE I
MEAN AND STANDARD DEVIATION FOR ALL CASES

		Case 1- 5000	Case 1- 20000	Case 2- 5000	Case 3- 5000
a	μ	5.8361e-005	$5.7811*10^{-5}$	$7.2793*10^{-4}$	0.0027
	σ	6.4567e-005	$6.3641*10^{-5}$	$6.6659*10^{-4}$	0.0026
b	μ	0.0033	0.0025	0.0077	0.0040
	σ	0.0060	0.0048	0.0112	0.0048
c	μ	0.3032	0.2610	0.4692	0.1935
	σ	0.0799	0.0690	0.0620	0.0518
d	μ	0.9320	0.9327	1.2642	0.6213
	σ	0.1220	0.1215	0.1325	0.0813

- The average distance to all the points, shown in d), remains the same as in Case 1-5000.

From the third observation, we conclude that the border is covered more densely when more points are used. From the fourth observation, we conclude that the area of the covered border has the same size as in Case 1-5000.

Because the above tests show that the area covered is the same and is almost independent of the number of points placed on the border, we implement the second case (i.e., penalizing the proximity of neighboring points). The objective is to examine

1) if there is an increase in the mean distance of each point to its closest neighbor with a decrease in the variance;
2) if the average distance of each point from the other points has increased.

From Fig. 6 and Table I we observe the following for Case 2 with 5000 points.

- The points, and midpoints are close to the border, but not as close as in the previous two tests, as shown in a) and b), respectively.
- There is a notable increase in the mean but a decrease in the variance of the distance to the closest neighbor, compared to Case 1-5000 and Case 1-20 000, as shown in c).
- The mean of the average distance of all points to all the other points has increased, compared to Case 1-5000 and Case 1-20 000, as shown in d).

From the first observation, we note that the points and midpoints are close to the border, but not as close as in Case 1-5000 or Case 1-20 000. The reason for this is that the fitness function now also emphasizes the separation of the points on the border, as in (3). From the second observation, we conclude that the points are distributed more uniformly and with greater spacing, hence covering a greater area, which is also verified by the third observation.

In Fig. 7 the results are summarized for better visualization, showing just the means for a), b), c), and d). In Fig. 7, we observe the following.

- Although the results of case 2 are satisfactory, the mean of the CTM of the midpoints, shown in b), is higher than in Case 1-5000 and Case 1-20 000.
- The distance to the closest neighbor, shown in c) is also higher than the previous two cases.

From these two observations, we conclude that the number of border points is not large enough. A remedy is to increase the number of points, but this requires an excessive amount of computational time. An alternative approach is to focus on the section of the border near the current operating point, which is done in Case 3.

From Fig. 6 and Table I we observe the following for Case 3 with 5000 points.

- The midpoints are much closer to the border than Case 2-5000, as shown in b).
- The points are further away from the border than all previous cases, as shown in a).
- The distance to the closest neighbor has decreased significantly, and has a smaller variance than Case 2-5000, as shown in c).
- The average distance of each point to all others has also decreased, as shown in d).

From the first and second observations, we conclude that the decrease in midpoint CTM is achieved while the CTM of the points is increased. However, their CTM is still acceptable. From the third observation, we conclude that the specific region of the border is covered more densely and uniformly. Finally, the fourth observation indicates that the points are concentrated in a specific section of the border closest to the current operating state.

The computation time required to compute 5000 points on the border is 1600 s using Matlab on a P4-1.7 GHz with 512 MB of RAM. However, our code is designed just for proof of principle and not optimized for speed. We maintain that by transferring the code to C or even to dedicated hardware, we can achieve improvements by several orders of magnitude. Furthermore, since the points are placed on the border in a sequential manner near the current operating condition, Case 3 can be used online.

VIII. Conclusion

In this paper, we propose a novel method for the identification of the security/vulnerability border of power systems. The technique is quite general, and can be applied to a number of problems where a desired border is required. The algorithm is excellent for real-time evaluation of the security index of a power system.

Acknowledgment

The authors would like to thank M. J. Healy, Boeing Company, and M. Kim, Electrical Engineering Department, University of Washington, for creating the training data.

References

[1] P. Kundur, *Power System Stability and Control*. New York: McGraw-Hill, 1994.
[2] H. Chiang, F. Wu, and P. Varaiga, “A BCU method for direct analysis of power system transient stability,” *IEEE Trans. Power Syst.*, vol. 9, pp. 1194–1208, Aug. 1994.
[3] C. K. Tang, C. E. Graham, M. El-Kady, and R. T. H. Alden, “Transient stability index from conventional time domain simulation,” *IEEE Trans. Power Syst.*, vol. 9, pp. 1524–1530, Aug. 1994.
[4] A. A. Fouad and V. Vittal, *Power System Transient Stability Analysis Using the Transient Energy Function Method*. Englewood Cliffs, NJ: Prentice-Hall, 1992.
[5] J. Sterngold, “California endures a 2nd straight day of power blackouts,” *The New York Times*, Jan. 19, 2001.
[6] Y. Mansour, “Competition and system stability—The reward and the penalty,” *Proc. IEEE*, vol. 882, pp. 228–234, Feb. 2000.
[7] L. Wehenkel, M. Pavella, E. Euxibie, and B. Heilbronn, “Decision tree based transient stability method: A case study,” *IEEE Trans. Power Syst.*, vol. 9, pp. 459–469, Feb. 1994.
[8] S. Rovnyak, S. Kretsinger, J. Thorp, and D. Brown, “Decision trees for real-time transient stability prediction,” *IEEE Trans. Power Syst.*, vol. 9, pp. 1417–1426, Aug. 1994.
[9] Y. Mansour, A. Y. Chang, J. Tamby, E. Vaahedi, and M. A. El-Sharkawi, “Large scale dynamic security screening and ranking using neural networks,” *IEEE Trans. Power Syst.*, vol. 12, pp. 954–960, May 1997.
[10] A. B. R. Kumar, A. Ipakchi, V. Brandwajn, M. El-Sharkawi, and G. Cauley, “Neural networks for dynamic security assessment of large-scale power systems: Requirements overview,” in *Proc. 1st Int. Forum Applicat. Neural Networks Power Syst.*, 1991, pp. 65–71.
[11] R. Marceau, R. Mailhot, and F. Galiana, “A generalized shell for dynamic security analysis in operations planning,” *IEEE Trans. Power Syst.*, vol. 8, pp. 1098–1832, Aug. 1993.
[12] K. Demaree, T. Athay, K. W. Cheung, Y. Mansour, E. Vaahedi, A. Y. Chang, B. R. Corns, and B. W. Garrett, “An online dynamic security analysis system implementation,” *IEEE Trans. Power Syst.*, vol. 9, pp. 1716–1722, Nov. 1994.
[13] R. Aresi, B. Delfino, G. B. Denegri, S. Massucco, and A. Morini, “A combined ANN/simulation tool for electric power system dynamic security assessment,” in *Proc. IEEE Power Eng. Soc. Summer Meeting*, vol. 2, 1999, pp. 1303–1309.
[14] D. J. Sobajic and Y.-H. Pao, “Artificial neural-net based dynamic security assessment for electric power systems,” *IEEE Trans. Power Syst.*, vol. 4, pp. 220–228, Feb. 1989.
[15] K. Omata and K. Tanomura, “Transient stability evaluation using an artificial neural network,” in *Proc. 2nd Int. Forum Applicat. Neural Networks Power Syst., ANNPS‘93 Conf.*, 1993, pp. 130–135.
[16] Y. Mansour, E. Vaahedi, and M. A. El-Sharkawi, “Dynamic security contingency screening and ranking using neural networks,” *IEEE Trans. Neural Networks*, vol. 8, pp. 942–950, July 1997.

[17] J. D. McCalley, S. Wang, Q. Zhao, G. Zhou, R. T. Treinen, and A. D. Papalexopoulos, “Security boundary visualization for systems operation,” *IEEE Trans. Power Syst.*, vol. 12, pp. 940–947, May 1997.

[18] C. A. Jensen, R. D. Reed, M. A. El-Sharkawi, and R. J. Marks, II, “Location of operating points on the dynamic security border using constrained neural network inversion,” in *Proc. Int. Conf. Intell. Syst. Applicat. Power Syst. (ISAP’97)*, Seoul, Korea, July 1997.

[19] R. D. Reed and R. J. Marks, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. Cambridge, MA: MIT Press, 1999.

[20] L. S. Moulin, M. A. El-Sharkawi, R. J. Marks, and A. P. Alves da Silva, “Automatic feature extraction for neural network based power systems dynamic security evaluation,” in *Proc. Int. Conf. Intell. Syst. Appl. Power Syst. (ISAP2001)*, Budapest, Hungary, July 2001.

[21] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[22] IEEE Committee Report, “Transient stability test systems for direct stability methods,” *IEEE Trans. Power Syst.*, vol. 7, pp. 37–44, Feb. 1992.

[23] R. D. Reed and R. J. Marks, II, “An evolutionary algorithm for function inversion and boundary marking,” in *Proc. IEEE Int. Conf. Evolutionary Computat.*, Nov. 26–30, 1995, pp. 794–797.

[24] R. Eberhart, “Particle swarm optimization,” in *Proceedings IEEE International Conference on Neural Networks (Perth, Australia)*. Piscataway, NJ: IEEE, 1995, vol. IV, pp. 1942–1948.

[25] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *The 1998 IEEE International Conference on Evolutionary Computation Proceedings, The 1998*. Piscataway, NJ: IEEE World Congress on Computational Intelligence, 1998, pp. 69–73.

[26] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in *Proc. 6th Int. Symp. Micro Machine Human Sci. (MHS’95)*, 1995, pp. 39–43.

[27] M. A. El-Sharkawi, Ed., *Evolutionary Techniques and Fuzzy Logic in Power Systems*. Piscataway, NJ: IEEE Press, 2000, 2000 TP-142.

[28] C. A. Jensen, M. El-Sharkawi, and R. J. Marks, “Power system security assessment using neural networks: Feature selection using Fisher discrimination,” *IEEE Trans. Power Syst.*, vol. 16, pp. 757–763, Nov. 2001.

Ioannis N. Kassabalidis (S’98) received the B.S. degree in electrical engineering from the Aristotle University of Thessaloniki, Greece, in 1998 and the M.S. degree in electrical engineering from the University of Washington, Seattle, in 2000, where he is currently pursuing the Ph.D. degree in electrical engineering.

His research interests include applications of biologically inspired algorithms and other computational intelligence topics.

Mohamed A. El-Sharkawi (F’95) was born in Cairo, Egypt, on April 28, 1948. He received the B.Sc. degree in electrical engineering from Cairo High Institute of Technology, in 1971 and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of British Columbia, Vancouver, BC, Canada, in 1977 and 1980, respectively.

In 1980, he joined the University of Washington, Seattle, as a faculty member where he is currently a Professor of electrical engineering. He is co-editor of the book *Neural Networks to Power Systems* (Piscataway, NJ: IEEE). He has published over 90 papers and book chapters. He holds five patents, three on adaptive var controller for distribution systems, and two on adaptive sequential controller for circuit breakers.

Dr. El-Sharkawi is the founder of the International Conference on the Application of Neural Networks to Power Systems (ANNPS) which was later merged with the Expert Systems Conference and renamed Intelligent Systems Applications to Power (ISAP).

Robert J. Marks, II (F’94) was born on August 25, 1950. He received the B.S. and M.S. degrees in electrical engineering from Rose-Hulman Institute of Technology, Terre Haute, IN, in 1972 and 1973, respectively, and the Ph.D. degree in electrical engineering from Texas Tech University, Lubbock, in 1977.

He is a Professor and Graduate Program Coordinator in the Department of Electrical Engineering, College of Engineering, University of Washington, Seattle. He is the author of numerous papers and is coauthor of the book *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks* (Cambridge, MA: MIT Press, 1999). He was a Topical Editor for optical signal processing and image science for the *Journal of the Optical Society of America*.

Dr. Marks is a Fellow of the Optical Society of America. He served as the first President of the IEEE Neural Networks Council. In 1992, he was given the honorary title of Charter President. He has served as the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS.

Luciano S. Moulin (S’98) received the B.Sc. and M.Sc. degrees in electrical engineering from the Federal Engineering School, Itajuba, Brazil, in 1995 and 1998, respectively, where he is currently pursuing the D.Sc. degree.

Alexandre P. Alves da Silva (SM’00) was born in Rio de Janeiro, Brazil, in 1962. He received the B.Sc. and M.Sc. degrees in electrical engineering from the Catholic University of Rio de Janeiro, Brazil, in 1984 and 1987, respectively, and the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1992.

He worked with the Electric Energy Research Center (CEPEL), from 1987 to 1988. During 1999, he was a Visiting Professor in the Department of Electrical Engineering, University of Washington, Seattle. Since 1993, he has been with the Federal Engineering School, Itajuba, where he is a Professor in electrical engineering. He and his group have developed intelligent forecasting and security assessment systems that are in operation at the control centers of Brazilian electric utilities. He has authored and coauthored about 150 papers on intelligent systems application to power systems.

Dr. Alves da Silva was the Technical Program Committee Chairman of the First Brazilian Conference on Neural Networks, in 1994, and of the International Conference on Intelligent System Applications to Power Systems, in 1999. He is the Chairman of the IEEE Neural Network Council Brazilian Regional Interest Group.