Multiple Element Contingency Screening

C. Matthew Davis, *Member, IEEE,* and Thomas J. Overbye, *Fellow, IEEE,*

***Abstract*—This paper presents a method for determining the double outage contingencies that threaten the system without solving the full contingency set. Two methods for contingency screening with complementary properties are presented. The results of the algorithms are compared to the full double outage contingency analysis results for a large North American case. The results show that the screening algorithms are able to detect nearly all of the contingencies that will result in violations, while requiring only a small fraction of the contingencies to be solved.**

***Index Terms*—contingency screening, contingency analysis, linear sensitivities.**

I. Introduction

MULTIPLE outage contingencies are becoming increasingly relevant because of the way the transmission grid is being used today. In the deregulated environment, the transmission system is utilized in ways the designers never contemplated, with the goal of operating the system as economically as possible. Combined with continual load growth, the change in system operations has resulted in a transmission system that is increasingly stressed. One way this is being dealt with is through the introduction of new standards requiring system operators to meet performance requirements in the event of multiple element outages [1].

While standards mandate utilities to consider multiple element outages [1], there are still technical challenges to overcome when processing the huge number of potential events. As multiple outages begin to be considered, the number of events to be considered grows rapidly as the number of outaged elements is considered. In particular,

$$list\ size = \binom{L}{k} = \frac{L!}{k!(L-k)!} \tag{1}$$

where L is the number of branches in the system and k is the number of outaged elements. For even a modestly sized system with 5000 branch elements, the number of double branch outage contingencies is almost 12.5 million.

For the single outage case ($k = 1$), the list size is simply L, which corresponds to the standard "n-1" criterion. For the double outage case ($k = 2$), the binomial coefficient can be expanded to

$$\binom{L}{2} = \frac{L(L-1)}{2} = \frac{L^2 - L}{2} \tag{2}$$

In order to evaluate the "n-2" security of the system, the number of events to be considered is on the order of L^2. Continuing along these lines, it is easy to show that for k simultaneous outages, $O(L^k)$ power flow solutions are required to process the contingency list. For power systems, the number of lines tends to scale linearly with the number of buses in the system, i.e., $L \approx 1.5 \cdot n$. Using this relation, we can convert $O(L^k)$ to a function of n

$$O(L^k) = O((1.5n)^k) = O((1.5)^k n^k) = O(n^k) \tag{3}$$

This conversion allows us to use the results in [2] to determine the computational order of processing multiple contingencies.

According to [2], solving the power flow with Newton's method requires $O(n^{1.4})$ computations and solving the power flow using linear methods requires $O(n^{1.2})$ computations. Combining these results with (3) gives the total computational order for solving k simultaneous outages. The computational order to solve multiple outage contingency analysis using Newton's method is

$$CE_N = O(n^{1.4}) \cdot O(n^k) = O(n^{k+1.4}) \tag{4}$$

and the computational order using linear methods is

$$CE_L = O(n^{1.2}) \cdot O(n^k) = O(n^{k+1.2}) \tag{5}$$

For the double outage case, which we will restrict ourselves to from this point forward, the computational order is $O(n^{3.4})$ when Newton's method is used and $O(n^{3.2})$ when linear methods are used. Thus, solving every double outage contingency is quite computationally intensive – to the point of being intractable for even relatively small systems – even when linear methods are used.

Using parallel computing techniques will speed up the process of solving contingencies. However, it does not address the problem of computational order. If the full list of contingencies is to be solved using parallel computing techniques, the speed-up will be proportional to the number of processors devoted to solving the list of contingencies (i.e., the amount of processing time needed is scaled by $\frac{1}{\text{\# of processors}}$). However, the list size will still be growing as given in (1). Thus, the fundamental issue is the computational order of the problem. For example, in the double outage case, the computational order remains $O(n^{3.2})$, since $O(n^{3.2})$ = $O(\alpha n^{3.2})$ for any constant α, including $\alpha = \frac{1}{\text{\# of processors}}$.

Contingency selection was developed as a method of determining which contingencies are important enough to add to the on-line contingency list [3]. The initial work on the topic used first order performance index sensitivities to rank contingencies [3]. Using performance index sensitivities has the advantage of being extremely fast. However, the results were found to be unreliable [4]. In an attempt to improve reliability, the use of higher order sensitivities was explored [5], [6]. Unfortunately, calculating the higher order sensitivities

The authors would like to acknowledge the support of the National Science Foundation under grant CNS-05-24695, the Power System Engineering Research Center (PSERC), and the Grainger Foundation

Tom Overbye is with the University of Illinois Urbana-Champaign, Urbana, IL 61801 (e-mail: overbye@illinois.edu). Matt Davis is with PowerWorld Corporation Champaign, IL 61821 (e-mail: matt@powerworld.com).

has no advantage over calculating the dc power flow [6]. A review of the early screening methods along with a derivation for calculating performance indices using linear sensitivities can be found in [7]. The work in [7] and [8] both store the inverse and handle multiple outages in the system, which is very fast. However, the storage requirements make handling large systems intractable.

Section II discusses the linear analysis that will be used to develop the screening methods for MW branch violations. Section III develops two screening methods. Section IV presents a sample application of the screening algorithm on the IEEE 14-bus case. Results for a large system are presented in Section V. Finally, conclusions and future work are presented in Section VI.

II. Linear Analysis Background

Linear sensitives are used to approximate real power flows for on-line power system operations because they are fast and reasonably accurate. In North America, flowgates are defined using linear sensitivities to monitor the post contingent state of transmission elements [9]. This section introduces the linear factors used throughout the rest of the paper, including a method of simulating multiple outages.

The dc assumptions convert the power flow into a linear problem [10]. The validity of the dc assumptions determines the accuracy of the linear approximations. Linear sensitivities are usually fairly accurate [11], [12]. However, their performance is constrained by the assumptions that underlie them. The dc assumptions are

- No resistive losses
- Bus voltages are 1.0 pu
- Angle differences across lines are small

When these assumptions are applied to the standard ac power flow equations, the familiar dc power flow equations are produced

$$\mathbf{B\Theta} = \mathbf{P} \tag{6}$$

where $\mathbf{B}$ is the dc Jacobian matrix, $\mathbf{\Theta}$ is a vector of bus angles, and the $\mathbf{P}$ is a vector of bus injections [13]. The linear system of equations can be used as a basis for the derivation of sensitivities, and in fact many useful sensitives have been developed taking this approach. However, because the dc assumptions include a flat voltage profile, they are unable to account for any changes in voltage. Correspondingly, the screening algorithms presented in this paper cannot account for voltage violations.

Power Transfer Distribution Factors (PTDFs) are the linear sensitivities of line flows (f) to a point-to-point transfer of power ($T_{(i,j)}$) [13], [14]

$$p_\alpha, (i,j) = \frac{\Delta f_\alpha}{T_{(i,j)}} = \frac{1}{x_\alpha}\mathbf{a}_\alpha \mathbf{B}^{-1}\mathbf{a}_{(i,j)}^T \tag{7}$$

where α denotes the line whose flow is being altered by the transfer; i and j are the buses where the injection and withdrawal take place. The symbol $\mathbf{a}_\alpha$ denotes a row vector containing a 1 and -1 at the from and to positions of line α, and $\mathbf{a}_{(i,j)}$ is a row vector containing a 1 and -1 at positions i and j respectively. In practice, PTDFs are used to approximate the change in flow on a transmission line caused by a change in injection and withdraw.

Line outage distribution factors (LODFs) are linear sensitivities of line flows to the preoutage flow on an outaged line β [13]

$$d_{\alpha,\beta} = \frac{\Delta f_{\alpha,\beta}}{f_\beta} = \frac{p_{\alpha,\beta}}{1 - p_{\beta,\beta}} \tag{8}$$

LODFs can be used to calculate the change in flow on line α after the outage of line β.

$$\Delta f_{\alpha,\beta} = d_{\alpha,\beta} f_\beta \tag{9}$$

Linear sensitivities have been extended to approximate changes due to multiple line outages [13] [15]. The formulation in [16] results in a simple matrix equation for post-outage flow. The trick to deriving the multiple outage expression is realizing that the outages affect each other. To account for this a system of equations can be constructed using the affected flows, which are written using tildes. For the outage of line β and line δ, the affected flows can be written as $\widetilde{f}_\beta$ and $\widetilde{f}_\delta$. Then, we can write a linear system of equations using the single outage sensitives, the preoutage flows, and the adjusted flows

$$\begin{aligned} \widetilde{f}_\beta &= f_\beta + d_{\beta,\delta}\widetilde{f}_\delta \\ \widetilde{f}_\delta &= f_\delta + d_{\delta,\beta}\widetilde{f}_\beta \end{aligned} \tag{10}$$

The system of equations states that adjusted flow on line β is a function of the pre outage flow on line β and the adjusted flow on line δ. It is possible to solve the system for the affected flows as long as the matrix is not singular, which occurs when $d_{\beta,\delta} = d_{\delta,\beta} = \pm 1$.

$$\begin{bmatrix} \widetilde{f}_\beta \\ \widetilde{f}_\delta \end{bmatrix} = \begin{bmatrix} 1 & -d_{\beta,\delta} \\ -d_{\delta,\beta} & 1 \end{bmatrix}^{-1} \begin{bmatrix} f_\beta \\ f_\delta \end{bmatrix} \tag{11}$$

These affected flows are the flows that the rest of the system would see if line β and line δ were outaged. Thus, these flows may be used for contingency analysis by multiplying by the appropriate LODF values. For example, the change in flow on line α may be expressed as

$$\Delta f_\alpha = [d_{\alpha,\beta}\ d_{\alpha,\delta}] \begin{bmatrix} \widetilde{f}_\beta \\ \widetilde{f}_\delta \end{bmatrix} \tag{12}$$

If we substitute (11) for the adjusted flows, we can arrive at a simple matrix equation for the change in flow on line α

$$\Delta f_\alpha = \mathbf{L}_\alpha \mathbf{M}^{-1}\mathbf{F} \tag{13}$$

where $\mathbf{M} \in \mathbb{R}^{2\times 2}$ is a matrix of LODF values relating the outaged lines to each other, $\mathbf{L}_\alpha \in \mathbb{R}^{1\times 2}$ is a row vector relating the outaged lines to the line α, and $\mathbf{F} \in \mathbb{R}^{2\times 1}$ is a vector of preoutage flows.

The matrix $\mathbf{M}$ compensates for multiple line outages. For the double line outage outage of line β and line δ, $\mathbf{M}$ can be expressed as

$$\mathbf{M} = \begin{bmatrix} 1 & -d_{\beta,\delta} \\ -d_{\delta,\beta} & 1 \end{bmatrix} \tag{14}$$

The vector $\mathbf{L}_\alpha$ is

$$\mathbf{L}_\alpha = [d_{\alpha,\beta}\ d_{\alpha,\delta}\ ] \tag{15}$$

and $\mathbf{F}$ contains the preoutage flows on lines β and δ

$$\mathbf{F}^T = [f_\beta \ f_\delta \ ] \tag{16}$$

It may be noted that the size of the linear system (10) can be increased to deal with more than two outages. However, this paper focuses on screening double outage contingencies, so the derivation is presented with that in mind.

III. Screening Algorithms

This section discusses two screening algorithms that are designed to generate a list of double outage contingencies which can be processed much faster than the complete list of double outage contingencies.

The fundamental assumption that the screening algorithms rely on is that the outage of a line only affects a small percentage of the other lines in the system. This assumption is unproven. However, in practice it holds for the power system models that we have encountered. Also, the failure of this assumption does not mean that important contingencies will be missed in contingency lists generated by the algorithms. It means that the output will be larger. If the outage of every line impacted every other line the same amount, then the algorithms would generate every double outage. Effectively, the algorithms in this section are recording the instances where the outage of a line has a large impact on other lines. The original inspiration for this work started with an examination of the matrix $\mathbf{M}$, which captures the impact of outaged lines on each other [17].

The impact tracking structure (ITS) algorithm is based only on sensitivity information, while the overload tracking structure (OTS) algorithm incorporates line flow and limit information as well. The algorithms have different strengths. The OTS algorithm is generally very good at detecting double outage contingencies that will result in violations. The ITS algorithm is good at detecting contingencies that only result in overloads when both lines are outaged, which tend to be difficult to detect.

Both screening algorithms are broken into two separate parts. The first part builds a structure that tracks the impact that lines have on each other. The second part of the algorithm uses the tracking structure to build a list of contingencies. Using different impact metrics and contingency generation algorithms allows the ITS and OTS to discriminate between different kinds of contingencies.

A. ITS Screening Algorithms

1) Impact Tracking Structure Construction: The impact tracking structure (ITS) is designed to track the impact of lines onto each other. As illustrated in Fig. 1, the structure is composed of a list of lists. Every branch in the system has a list that contains the lines whose outage has the ability to change the flow on that branch as measured using LODFs. Construction of the ITS requires the computation of L^2 LODF values, which has a computational order of $O(n^{2.2})$.

Algorithm 1 gives the method for constructing the impact tracking structure. The algorithm is very similar to single outage contingency analysis using LODFs. LODF values for every single outage are calculated, and the values above the threshold, d^*, are recorded. The result is a structure where every line in the system is associated with a list of lines that impact it.

The threshold, d^*, is a an input parameter that specifies a lower bound for recording an impact. Since values below d^* are not added to the ITS, varying the value of d^* will affect the size of the ITS. The larger the value of d^*, the fewer entries are added to the ITS. The smaller the value of d^*, the more entries will be made in the ITS. In the most extreme case, d^* is zero. In this case, every one of the L^2 LODF values will be recorded. This will result in an extremely large ITS and the algorithm to generate the contingency list will become extremely slow (although, the result can be anticipated: The algorithm will simply return every double outage contingency).

To balance the speed of the algorithm with the completeness of the results, engineering judgment, knowledge of the system, and the study requirements must be used to choose an appropriate value for d^*. For the studies presented in this paper, the NERC flowgate cutoff of 5% [18] was used as a lower bound for d^*. For a well-chosen value of d^*, the ITS is a very sparse structure, which reflects the fact that the outage of a line typically only affects a few lines in a large system.

The value of d^* is system specific, meaning that each system will have its own best value. This means that picking a good value requires some initial work. Picking the threshold to yield the largest contingency list that can be solved in the desired amount of time is a good approach because it covers as many potential events as possible. In our experience experimenting with threshold values, the size of the tracking structures grows nearly exponentially once the threshold value gets below a certain value. This indicates that many lines are being added to the tracking structure because the threshold is not filtering them out. As a practical matter, starting with smaller threshold values and increasing them until the output contingency list reaches the desired size is a good way to determine the threshold value.

Input: List of lines, LODF threshold d^*
Output: ITS
foreach *Line* α **do**
  foreach *Line* β **do**
    if $\alpha \neq \beta$ **then**
      Calculate $d_{\alpha,\beta}$ **if** $|d_{\alpha,\beta}| \geq d^*$ **then**
        Add entry at Row α for line β;
      end
    end
  end
end

Algorithm 1: ITS construction algorithm

An illustration of the tracking structure is shown in Figure 1. This figure illustrates how each line is associated with a list of lines, which have been added based on their LODF values. Only lines whose impact is larger than the threshold are added to the list. For example, the first row shows that line 1 is only impacted by three other lines (line 2, line 3, and line 4). This means that only these three lines impact line 1 above the cutoff

threshold. While the outage of other lines in the system would impact the flow on line 1, the amount of the impact – measured using LODF values – is small.

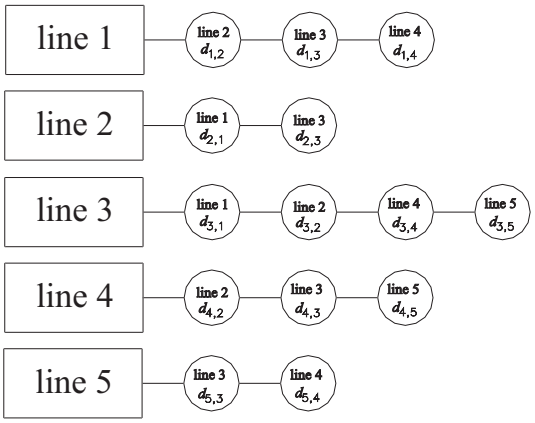

Fig. 1. Impact tracking structure

2) Contingency List Generation: Once the ITS has been constructed, contingency selection is a two-step process. First, every possible double outage contingency is generated for each row of the ITS. Second, the non-unique outages are removed from the list. When the algorithm terminates, we have a list of flagged contingencies (FL) where every contingency generated by the process involves lines that both impact any third lines. The algorithm for generating FL from the ITS is given in Algorithm 2.

Input: Impact Tracking Structure ITS
Output: List of Flagged Contingencies FL
foreach *Row r in the ITS* **do**
  Generate all combinations of the elements in r;
  Add combinations to FL;
end
Remove non-unique elements from FL.

Algorithm 2: ITS list generation algorithm

For the example ITS shown in Fig. 1, generating the combinations for the first row would generate

$$\binom{3}{2} = 3 \tag{17}$$

double outage contingencies. For the entire structure, generating the combinations of each row results in 14 double outage contingencies. However, these 14 contingencies are not unique because the same contingencies show up more than once. For example, the contingency involving line β and line γ is generated for the first and third rows. This means we must remove the non-unique elements from the flagged list.

Removing the non-unique elements from the list FL is the final step in the screening algorithm. There are several approaches to remove the non-unique elements. However, care must be taken to choose an efficient method. A naive approach can easily result in a factorial time algorithm.

One efficient approach is to sort the list, then pass through once recording unique elements. Sorting is a heavily studied problem, and many efficient sorting algorithms exist [19]. The final pass-through in which the non-unique elements are removed is linear time.

Another more efficient approach is to use data structures that only allow insertion of unique elements. This type of data structure is maintained in a sorted state, so the sorting and comparing mentioned above is automatically performed upon insertion to the structure. When an item is added, a binary search is used to determine if the item already exists in the structure. If the element is not already in the structure, it is added. Otherwise, it is discarded. The searching can be done efficiently because the structure is maintained in a sorted state. This kind of structure is known as a sorted associative container [20]. A standard implementation of this type of container is distributed as part of the standard template library (STL) [21], which provides much of the functionality of the C++ language.

To examine the computational effort of generating the list of flagged contingencies, we need to know the average row length of the ITS because it will determine the number of outages generated by the combinations which in turn determines the number of insertion operations needed. If we define average row length to be R, then generating every double outage combination has a computational effort $O(R^2)$. Insertion into a sorted associative container is $O(log(R))$ in the worst case [20], and there will be R^2 insertions. This gives a computational complexity of $O(log(R^2))$, which reduces to $O(log(R^2)) = O(2log(R))$.

In practice, the computational effort to run Algorithm 2 depends on the choice of d^*, which is ultimately what determines R. Since there is no simple relationship between d^* and R, the runtime of the list generation algorithm is difficult to quantify. However, in practice, the list generation phase is very fast.

B. OTS Screening Algorithm

The second screening algorithm takes advantage of more information about the system. The overload tracking structure (OTS) screening algorithms use line limit and flow information in addition to the sensitivity information that was used to construct the ITS. Instead of measuring impacts using only sensitiviy information, the OTS construction uses post contingent flows calculated using sensitiviy information ($\Delta f_\alpha = d_{\alpha,\beta} f_\beta$). This makes the OTS constuction slightly more complex. However, it also makes the screening results very accurate.

1) OTS Construction: The overload threshold value, o^* is the key parameter in the OTS construction algorithm, given in Algorithm 3. The overload threshold is a margin away from a single outage post contingent overload. For example, specifying an overload threshold value of 5% means that an outage that results in a flow of 95% of rated limit will be included in the OTS. The computational order of the OTS construction algorithm is the same as the ITS construction algorithm because the dominate term is still the calculation of the sensitivites.

2) Contingency List Generation: The overload tracking structure uses a different algorithm for generating the list

Input: List of lines
Input: Flow change threshold o^*
Input: Line Flows
Output: OTS
foreach *Line* α **do**
  foreach *Line* β **do**
    if $\alpha \neq \beta$ **then**
      Calculate $d_{\alpha,\beta}$; Calculate Δf_α; **if** $\Delta f_\alpha \geq o^*$ **then**
        Add entry at Row α for line β;
      end
    end
  end
end

Algorithm 3: OTS construction algorithm

of contingencies from the tracking structure. The motivation behind using a different tracking structure is the desire to capture more contingencies. The algorithm is given in Algorithm 4. This algorithm generates a list of contingencies by combining the entries in the tracking structure with every other line in the system. This is designed to capture the single outage contingencies that result in violations when they are paired with other lines in the system. The OTS contingency list generation algorithm uses the same technique to remove the redundant contingencies from the list it generates.

Input: Overload Tracking Structure OTS
Output: List of Flagged Contingencies FL
foreach *Row r in the ITS* **do**
  foreach *Element e in row r* **do**
    Generate combination of e with every other line in the system
  end
end
Remove non-unique elements from FL.

Algorithm 4: OTS list generation algorithm

The OTS contingency list generation algorithm has the potiental to be slower than the ITS contingency list generation algorithm because combining the elements in the tracking structure with every line in the system can result in a large number of contingencies. However, the large-system results indicate that for practical systems the OTS tends to be very small (i.e., each row has very few entries). This means that for pratical systems, the OTS contingency list generation process is very fast.

IV. Tracking Structure Construction Examples

The IEEE 14-bus test case [22] was chosen to illustrate the tracking structure construction algorithms because it is small enough to be presentable and large enough that the tracking structures are interesting. The IEEE 14-bus test case has 20 lines, which are given unique names in Fig. 2. These names are used to identify lines in the tracking structures. They are also used in Table I to identify lines in the LODF matrix.

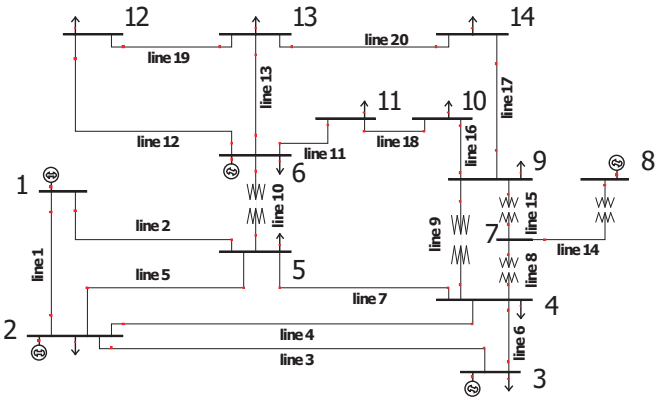

Fig. 2. IEEE 14-bus one-line diagram

A. ITS Generation Example

The algorithms presented in section III were applied to the IEEE 14-bus test case system. The first two rows of the impact tracking structure that is generated are shown below in Fig. 3. For this particular structure the LODF cutoff, d^*, is 10%.

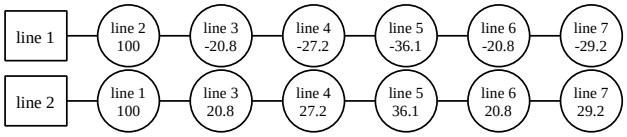

Fig. 3. First two rows of the IEEE 14-bus ITS

The ITS construction algorithm can be illustrated using LODF values for the IEEE 14-bus system given in Table I. The first row, corresponding to line 1, contains the LODF values for the outage of lines 2, 3, 4, 5, 6, and 7. For example, row 1, column 3 contains $d_{line\ 3,\ line\ 1}$. The algorithm works by proceeding down the row and adding an entry to the list corresponding to the column every time the LODF value exceeds the threshold, $d^* = 10\%$. Thus, for the first row of Table I, there are entries made in the lists of line 2, line 3, line 4, line 5, line 6, and line 7. When this process is repeated for every line, the final product is the ITS. The first two rows of the ITS for the IEEE 14-bus case with $d^* = 10\%$ are shown in Fig. 3.

B. OTS Generation Example

The OTS for the IEEE 14 bus test case is shown in Fig. 4. The OTS was generated using an overload threshold value (o^*) of 10%. The rows of the OTS contain a line identifier and the post contingent flow as a percentage of the line's limit that was calcluated using linear sensitivites. Since there are no line limits in the IEEE 14 bus test case, a limit of 100 MW was used for every line. The circles contain a line identifier and the predicted change in flow for the outage of the line identified in the circle. In other words, when the line identified in the circle is outaged, the post contingent flow on the line that the row belongs to is the other value in the circle. The overload threshold value, o^*, means that values less than 100% can be included in the OTS, which explains the entries in the 90s on the second row.

TABLE I
IEEE 14-BUS LODF MATRIX

	line 1	line 2	line 3	line 4	line 5	line 6	line 7	line 8	line 9	line 10	line 11	line 12	line 13	line 15	line 16	line 17	line 18	line 19	line 20
line 1	**-100.0**	**100.0**	**-16.9**	**-35.3**	**-47.8**	**-16.9**	**-49.4**	-1.8	-1.0	2.8	1.7	0.3	0.9	-1.8	-1.7	-1.1	-1.7	0.3	1.1
line 2	**100.0**	**-100.0**	**16.9**	**35.3**	**47.8**	**16.9**	**49.4**	1.8	1.0	-2.8	-1.7	-0.3	-0.9	1.8	1.7	1.1	1.7	-0.3	-1.1
line 3	**-20.8**	**20.8**	**-100.0**	**45.5**	**33.7**	**-100.0**	**-51.5**	-1.9	-1.1	2.9	1.8	0.3	0.9	-1.9	-1.8	-1.2	-1.8	0.3	1.2
line 4	**-27.2**	**27.2**	**28.6**	**-100.0**	**44.2**	**28.6**	**-67.6**	-2.4	-1.4	3.8	2.3	0.3	1.2	-2.4	-2.3	-1.5	-2.3	0.3	1.5
line 5	**-36.1**	**36.1**	**20.7**	**43.3**	**-100.0**	**20.7**	**60.5**	2.2	1.3	-3.4	-2.1	-0.3	-1.1	2.2	2.1	1.4	2.1	-0.3	-1.4
line 6	**-20.8**	**20.8**	**-100.0**	**45.5**	**33.7**	**-100.0**	**-51.5**	-1.9	-1.1	2.9	1.8	0.3	0.9	-1.9	-1.8	-1.2	-1.8	0.3	1.2
line 7	**-29.2**	**29.2**	**-24.8**	**-51.8**	**47.4**	**-24.8**	**-100.0**	**14.9**	8.6	**-23.4**	**-14.1**	-2.1	-7.3	**14.9**	**14.1**	9.3	**14.1**	-2.1	-9.3
line 8	-2.9	2.9	-2.5	-5.2	4.8	-2.5	**41.5**	**-100.0**	**50.8**	**49.2**	**29.6**	4.4	**15.2**	**-100.0**	**-29.6**	**-19.6**	**-29.6**	4.4	**19.6**
line 9	-2.1	2.1	-1.8	-3.8	3.5	-1.8	**30.1**	**64.3**	**-100.0**	**35.7**	**21.5**	3.2	**11.1**	**64.3**	**-21.5**	**-14.2**	**-21.5**	3.2	**14.2**
line 10	6.0	-6.0	5.1	**10.6**	-9.7	5.1	**-84.4**	**63.5**	**36.5**	**-100.0**	**-60.2**	-8.8	**-30.9**	**63.5**	**60.2**	**39.8**	**60.2**	-8.8	**-39.8**
line 11	3.6	-3.6	3.0	6.3	-5.8	3.0	**-50.2**	**37.8**	**21.7**	**-59.5**	**-100.0**	9.0	**31.5**	**37.8**	**100.0**	**-40.5**	**100.0**	9.0	**40.5**
line 12	0.4	-0.4	0.3	0.7	-0.6	0.3	-5.5	4.1	2.4	-6.5	6.7	**-100.0**	**86.8**	4.1	-6.7	**13.2**	-6.7	**-100.0**	**-13.2**
line 13	1.0	-1.0	0.9	1.8	-1.7	0.9	**-14.4**	**10.8**	6.2	**-17.1**	**17.6**	**65.4**	**-100.0**	**10.8**	**-17.6**	**34.6**	**-17.6**	**65.4**	**-34.6**
line 15	-2.9	2.9	-2.5	-5.2	4.8	-2.5	**41.5**	**-100.0**	**50.8**	**49.2**	**29.6**	4.4	**15.2**	**-100.0**	**-29.6**	**-19.6**	**-29.6**	4.4	**19.6**
line 16	-3.6	3.6	-3.0	-6.3	5.8	-3.0	**50.2**	**-37.8**	**-21.7**	**59.5**	**100.0**	-9.0	**-31.5**	**-37.8**	**-100.0**	**40.5**	**-100.0**	-9.0	**-40.5**
line 17	-2.9	2.9	-2.5	-5.2	4.8	-2.5	**41.5**	**-31.3**	**-18.0**	**49.2**	**-50.8**	**22.2**	**77.8**	**-31.3**	**50.8**	**-100.0**	**50.8**	**22.2**	**100.0**
line 18	-3.6	3.6	-3.0	-6.3	5.8	-3.0	**50.2**	**-37.8**	**-21.7**	**59.5**	**100.0**	-9.0	**-31.5**	**-37.8**	**-100.0**	**40.5**	**-100.0**	-9.0	**-40.5**
line 19	0.4	-0.4	0.3	0.7	-0.6	0.3	-5.5	4.1	2.4	-6.5	6.7	**-100.0**	**86.8**	4.1	-6.7	**13.2**	-6.7	**-100.0**	**-13.2**
line 20	2.9	-2.9	2.5	5.2	-4.8	2.5	**-41.5**	**31.3**	**18.0**	**-49.2**	**50.8**	**-22.2**	**-77.8**	**31.3**	**-50.8**	**100.0**	**-50.8**	**-22.2**	**-100.0**

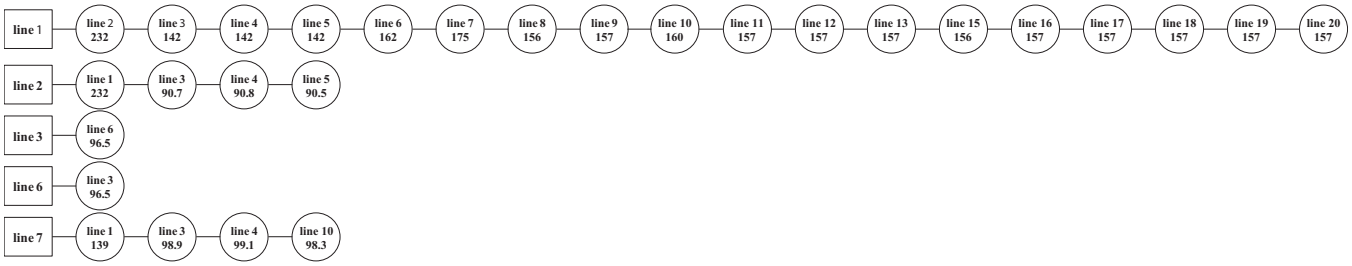

Fig. 4. OTS for the IEEE 14 bus test case

Of course, after generating the tracking structures the next step is to generate the list of contingencies. For the ITS, this means generating a contingency list by taking the combinations of the entries in each row. For the OTS, the contingency list is generated by taking the combinations of the entries in the OTS with the other lines in the system.

V. SCREENING RESULTS

The screening algorithms in Section III were each applied to a 5395 bus, 7616 line system based on a large North American utility. Considering every double line outage in the system results in a contingency list with 28,997,920 events. The results of the screening algorithms are compared against the full double outage contingency results, solved using the dc power flow. The power flow solutions found a total of 546,557 contingencies that result in violations. Violations are recorded whenever a line was loaded over 100% of its limit. There are 1851 violations that only result from the outage of two lines. These 1851 contingencies are grouped in a list called the double violation list. These contingencies are important because they are difficult to detect based on the single-outage LODF information that forms the basis for the ITS and OTS algorithms).

A. OTS Screening Results

The results in Table II show that the OTS algorithms do a very good job of predicting the double outage contingencies that will result in violations. It does this by using a padded version of the single outage contingency analysis results, and predicting that a single outage that creates a violation will also create a double outage violation. The contingency list generated using an o^* of 0 results in a list with 598,504 contingencies. Solving these 598,504 contingencies captures over 99% of the double outages with violations. This is only 2.06% of the total number of double outage contingencies. As the padding is increased, the number of captured contingencies increases. However, beyond a certain point the number of extra contingencies explodes. As can be seen by examining the last two rows in Table II, changing the value of o^* from 0.050 to 0.075 resulted in the number of extra contingencies increasing by a factor of ten. The reason for the explosion is that there are many more entries in the OTS when the threshold is increased. There is only one missed contingency, however capturing this contingency comes at the expense of creating a contingency list that contains nearly all of the double outage contingencies.

TABLE II
FULL VIOLATION LIST OTS SCREENING RESULTS

o^*	captured	extra	missed	captured (%)
0.000	546557	51947	1562	99.72
0.025	547025	126794	1094	99.8
0.050	548118	276683	653	99.88
0.075	548118	27739966	1	99.9998

Detecting double contingencies that only result in violations when both lines are out is an area of relative weakness for the OTS screening algorithms, as indicated by the results in Table III. Fortunately, detecting this type of contingency is what the ITS screening algorithm is best at.

TABLE III
DOUBLE VIOLATION LIST OTS SCREENING RESULTS

o^*	captured	extra	missed	captured (%)
0.000	0	598504	1851	0
0.025	468	673351	1383	25.28
0.050	909	823240	942	49.11
0.075	1555	28286529	296	84.0

B. ITS Screening Results

The results of the ITS screening algorithm are presented for 5 different values of d^*. The minimum value was selected to be 5%, based on the NERC threshold for transaction curtailment [18]. The threshold value is varied from the minimum 5% to 25% in increments of 5%. This is done to examine the behavior of the screening algorithm as its key parameter is varied.

The experimental results show that the screening algorithms screen out a very high percentage of the double outage events. That is, the algorithms do not generate an excessively large list. Unfortunately, the algorithm fails to capture a large number of severe contingencies.

Table IV compares the screening results for the ITS screening algorithms, comparing them against the full list of severe contingencies. The table contains the number of severe contingencies captured, the number of extra (non threatening) contingencies generated by the algorithm, the number of missed severe contingencies, and the percentage of contingencies captured. The percentage is calculated by dividing the number captured by the total number of severe contingencies. For example, the 6.6% captured for the d^* value of 0.05 is calculated as

$$\%captured = 100 \cdot \frac{36,153}{546,557} = 6.6\% \tag{18}$$

Examining Table IV, it is clear that the ITS based algorithms fail to capture a large number of severe contingencies. For the most conservative value of d^* only 6.6% of the severe contingencies are captured.

TABLE IV
FULL VIOLATION LIST ITS SCREENING RESULTS

d^*	captured	extra	missed	captured (%)
0.05	36153	1374307	511966	6.6
0.10	13877	514485	534242	2.5
0.15	7038	274220	540181	1.3
0.20	4790	168124	543329	0.87
0.25	2909	113858	545210	0.053

The results of the ITS screening algorithms are compared to the double violation list in Table V. These results show more promise. The algorithms captures between 30.6% and 66.5% of the contingencies in the double violation list. While these values may not sound particularly high, capturing the contingencies in the double violation list is difficult, and the ITS algorithm has the best performance in this area. The reason that the ITS performs well at generating contingencies in the double violation list is the way that the list of contingencies is generated from the ITS. The list is generated by taking the combinations of the elements in the ITS rows, and this process generates a list of double outage contingencies in which each line of a contingency impacts a line above the d^* threshold.

TABLE V
DOUBLE VIOLATION LIST ITS SCREENING RESULTS

d^*	captured	extra	missed	captured (%)
0.05	1230	1409230	621	66.5
0.10	1016	527346	835	54.9
0.15	785	280473	1066	42.4
0.20	653	172261	1198	35.3
0.25	567	116200	1284	30.6

C. Tracking Structure Statistics

Along with the results from the screening algorithms, information about the tracking structures was collected. The information includes the size (i.e., the number of non-zero entries), the number of rows with at least one entry, the number of rows with no entries, the maximum length of any row, and the average row length. The average row length is calculated by dividing the number of entries by the total number of rows. The total number of rows is equal to the number of lines in the system since there is a row for each line in the system. Also, the number of non-zero rows and the number of zero-length rows add to equal the number of lines in the system.

1) ITS Size Statistics: The size data for the ITS is given in Table VI. The size information shows that the larger the value of d^*, the fewer entries are made in the ITS. This makes sense because raising the value of d^* means that a line must have a greater LODF value to be entered into the tracking structure. The size decreases very rapidly as the threshold, d^*, increases. The average row and maximum row length also decrease rapidly.

TABLE VI
ITS SIZE DATA

d^*	ITS size	avg. row length	max. row length	non-zero rows
5%	284507	37	339	6326
10%	150853	19	180	6191
15%	100105	13	96	6099
20%	72260	9	67	6018
25%	55560	7	49	5929

2) OTS Size Statistics: Information about the size of the OTS is shown in Table VII. The size information shows that it has some unique properties when compared to the other tracking structure. First, the size of the OTS increases as the amount of padding, o^*, increases. This is because the more padding is added, the more entries are made in the tracking structure. Also, the size of the OTS is very small. There are only 13,056 entries in the largest case. Examining the number

of non-zero rows, it can be seen that the entries in the OTS appear at only a few rows.

TABLE VII
OTS SIZE DATA

o^*	OTS size	avg. row length	max. row length	non-zero rows
0.000	113	0.015	35	48
0.025	130	0.017	38	56
0.050	159	0.021	49	61
0.075	13056	1.710	6424	76

VI. CONCLUSION

This paper presents a method of screening contingencies in order to avoid the computational expense of processing every double outage event. Two algorithms are presented. One uses only sensitivity information to generate a list of severe contingencies, and the other uses sensitivity, flow, and limit information. The results of the screening algorithms are compared to the contingency analysis results for a large North American case. The ITS screening algorithms are good at detecting contingencies that result in violations only when both lines are outaged. The OTS screening algorithms are very good at detecting severe contingencies in general.

REFERENCES

[1] NERC. (2005) System performance following loss of two or more bulk electric system elements (category c). [Online]. Available: www.nerc.com/files/TPL-003-0.pdf

[2] F. L. Alvarado, "Computational complexity in power systems," *IEEE Transactions on Power Apparatus and Systems*, vol. 95, no. 4, pp. 1028–1037, Jul. 1976.

[3] G. C. Ejebe and B. F. Wollenberg, "Automatic contingency selection," *IEEE Transactions on Power Apparatus and Systems*, vol. 98, no. 1, pp. 97–109, Jan. 1979.

[4] G. Irisarri, D. Levner, , and A. Sasson, "Automatic contingency selection for on-line security analysis - real-time tests," *IEEE Transactions on Power Apparatus and Systems*, vol. 98, no. 5, pp. 1552–1559, Sep. 1979.

[5] T. Mikolinnas and B. Wollenberg, "An advanced contingency selection algorithm," *IEEE Transactions on Power Apparatus Systems*, vol. 100, no. 2, pp. 608–617, Feb. 1981.

[6] G. Irisarri and A. Sasson, "An automatic contingency selection method for on-line security analysis," *IEEE Transactions on Power Apparatus and Systems*, vol. 100, no. 4, pp. 1838–1844, Apr. 1981.

[7] B. Stott, O. Alsaç, and F. Alvarado, "Analytical and computational improvements in performance-index ranking algorithms for networks," *Int J of Electrical Power and Energy Systems*, vol. 7, no. 3, pp. 154–160, Jul. 1985.

[8] M. K. Enns, J. J. Quada, , and B. Sackett, "Fast linear contingency analysis," *IEEE Transactions on Power Apparatus and Systems*, vol. 101, no. 4, pp. 783–791, Apr. 1982.

[9] NERC. (2008) Nerc operating manual. [Online]. Available: www.nerc.com/files/opman_12-13Mar08.pdf

[10] B. Stott and O. Alsac, "Fast decoupled load flow," *IEEE Transactions on Power Apparatus and Systems*, vol. 93, no. 3, pp. 859–869, May 1974.

[11] R. Baldick, "Variation of distribution factors with loading," *IEEE Transactions on Power Systems*, vol. 18, no. 4, pp. 1316–1323, 2003.

[12] M. Liu and G. Gross, "Effectiveness of the distribution factor approximations used in congestion modeling," in *Proc. 14th Power System Computational Conference*, Sevilla, Spain, jun 2002.

[13] A. J. Wood and B. F. Wollenberg, *Power Generation Operation and Control*. John Wiley and Sons, 1996.

[14] NERC, *Transmission transfer capability: a reference document for calculating and reporting the electric power transfer capability of interconnected electric systems*. NERC, 1995.

[15] T. Guler and G. Gross, "Detection of island formation and identification of causal factors under multiple line outages," *IEEE Transactions on Power Systems*, vol. 22, no. 2, pp. 505–513, May 2007.

[16] C. M. Davis, T. J. Overbye, and J. D. Weber, "Index of cascadeability using linear contingency evaluation," in *Proc. North American Power Symposium*, Moscow, ID, Aug. 2004.

[17] C. Davis and T. Overbye, "Linear analysis of multiple outage interaction," in *Hawaii International Conference on System Science*, Jan. 2009.

[18] NERC. (2006) Nerc iro-006-03 reliability coordination transmission loading relief. [Online]. Available: www.nerc.com/files/IRO-006-3.pdf

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. McGraw Hill, 2001.

[20] D. R. Musser, G. J. Derge, and A. Saini, *STL Tutorial and reference guide*, 2nd ed. Addison Wesley, 2001.

[21] SGI. (2007) Standard template library programmer's guide. [Online]. Available: http://www.sgi.com/tech/stl/index.html

[22] U. of Washington. (2007) Power system test case archive. [Online]. Available: http://www.ee.washington.edu/research/pstca/

C. M. Davis (S'03-M'05) received the B.S. degree in electrical engineering from Louisiana Tech University in 2002 and the M.S. and Ph.D. degrees from the University of Illinois Urbana-Champaign. Currently he is working at PowerWorld Corporation. His research interests include linear sensitivities, power system analysis, power system visualization, and power system operational reliability.

T. J. Overbye (S'87-M'92-SM'96-F'05) received the B.S., M.S. and Ph.D. degrees in electrical engineering from the University of Wisconsin-Madison. He is currently the Fox Family Professor of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He was with Madison Gas and Electric Company, Madison, WI, from 1983-1991. His current research interests include power system visualization, power system analysis, and computer applications in power systems.