Application of Cray XMT for Power Grid Contingency Selection

Yousu Chen, **Shuangshuang Jin**, **Daniel Chavarría-Miranda**, *and* **Zhenyu Huang**
Pacific Northwest National Laboratory

ABSTRACT: *Contingency analysis is a key function to assess the impact of various combinations of power system component failures. It involves combinatorial numbers of contingencies that exceed the capability of even very large supercomputing platforms. Therefore, it is critical to select credible contingency cases within the constraint of computing power. This paper presents a contingency selection method employing graph theory (edge betweenness centrality) to analyze power grid weighted graphs to remove low-impact components. The parallel implementation of the method was successfully demonstrated on the Cray XMT machine. The implementation takes advantage of the superior capabilities of the Cray XMT for graph-based problems and its programming features. This paper presents the performance and scalability of the Cray XMT and comparison with a cache-based, scalable, shared-memory machine.*

KEYWORDS: Contingency selection, betweenness centrality, parallel computing.

1. Introduction

Contingency analysis is a key function in the Energy Management System (EMS), which assesses the ability of the power grid to sustain various combinations of power grid component failures based on state estimates. Because of the heavy computation involved, today's contingency analysis can be updated only every few minutes for only a select set of "N-1" contingency cases (i.e., failures of any one component). As electricity demand continues to grow and renewable energy increases its penetration in the power grid, the personnel operating and managing the power grid are facing some fundamental challenges. Contingency analysis evolves from previous "N-1" analysis to "N-x" analysis (i.e., failures of multiple components). The combinatorial number of contingency cases increases exponentially as "x" becomes larger. For large systems, the contingency analysis process imposes a substantial computational burden. For example, for the western US power grid, which includes 17,000 branches, the case number could easily reach 10^{21} for "N-5" contingency analysis. This results in many more cases to analyze and much more data from the analysis to present to power grid operators. Because most of the contingency cases have a low probability to occur and/or have low impact on the system, contingency selection methods need to consider the probability and impact in order to identify a credible smaller set of cases for further vulnerability assessment.

In the past several decades, some research has been conducted in the area of contingency selection. The previous research includes performance indices (PI) related contingency ranking method based on approximate power flow solutions[1],[2], contingency evaluation using concentric relaxation [3], sparse vector methods [4], partial refactorization method [5] , bounding method for AC contingency analysis[6], hybrid method[7] and quadratized power flow sensitivity analysis[8]. These existing methods are of various qualities in identifying the credible set of contingency cases. From the computational point of view, many of these methods still involve some kind of simplified analysis of all contingency cases. The methods may be feasible for "N-1" contingency analysis. However, for "N-x" analysis, the sheer number of cases leads to the impracticality of even the simplified computation for all cases. We must search for a more efficient contingency

selection method.

This paper proposes a new contingency selection method based on the concept of graph edge betweenness centrality. Power grid can be treated as a weighted undirected graph, and the edge betweenness centrality identifies the most "traveled" edges in the graph. The most traveled edges are considered the most important branches in a power grid, and their failures must be analyzed, while the least traveled edges are identified as low-impact branches, and their failures do not need to be analyzed because they are of little importance to power grid stability.

This paper starts with a description of the implementation of edge betweenness centrality using a modified version of Brandes' algorithm [12] on the Cray XMT in Section 2, followed by Section 3 describing the case study for a 760-bus power system, which shows the result of the proposed contingency selection method validated with the actual impact of power grid component failures based on performance indices. The Cray XMT performance analysis and the performance comparison with a cache-based, scalable, shared-memory machine are presented in Sections 4 and 5. Section 6 concludes the paper with suggested future work.

2. Betweenness Centrality implementation

2.1 Edge Betweenness Centrality Definition

Vertex betweenness is a centrality measure of a vertex within a graph. The definition of vertex betweenness centrality is defined as eq. 1: [9],[10]

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad \text{(eq. 1)}$$

where

$C_B(v)$ = vertex betweenness,

σ_{st} = the number of shortest paths from any two vertices $s \in V$ to $t \in V$ in a graph, and

$\sigma_{st}(v)$ = the number of shortest paths from s to t via vertex v.

Edge betweenness is a centrality measure of an edge within a graph. Edges that occur on many shortest paths between any vertex pair have higher betweenness than those that do not. The following eq. 2 is the standard measure of edge betweenness centrality [11]:

$$C_{eB}(e) = \sum_{s \neq t \in V} \frac{\sigma_{st}(e)}{\sigma_{st}} \qquad \text{(eq. 2)}$$

where

$C_{eB}(e)$ = edge betweenness,

σ_{st} = the number of shortest paths from any two vertices $s \in V$ to $t \in V$ in a graph $G = (V, E)$, and

$\sigma_{st}(e)$ = the number of shortest paths from s to t that some edge $e \in E$ lies one.

2.2 Modified Brandes' algorithm

Ulrik Brandes' algorithm [12] is frequently used to calculate vertex betweenness for unweighted/weighted graphs. It requires $O(n+m)$ space and $O(nm)$ time for unweighted graphs, $O(nm + n^2 \log n)$ time for weighted graphs, where n and m are the number of vertices and edges in the graph, respectively.

In [12], the pseudo code for calculating vertex betweenness for unweighted graphs was provided. In order to apply edge betweenness centrality to power grid graphs, we adapted Brandes' algorithm to calculated edge betweenness for weighted power grid graphs. In power grids it is possible that there are multiple shortest paths between two buses, therefore the Dijkstra's algorithm [13] is also adapted to store multiple shortest paths.

The pseudo code for the adapted Brandes' algorithm in weighted graphs is as follows:

```
C_vB[v] ← 0, v ∈ V;   // vertex betweenness array
C_eB[e] ← 0, e ∈ E;   // edge betweenness array
visited[v] ← 0, v ∈ V;
for s ∈ V do
  S ← empty stack;
  P[w] ← empty list, w ∈ V ;
  σ[t] ← 0, t ∈ V; σ[s] ← 1;
  d[t] ← 0, t ∈ V; d[s] ← 0;
  Q ← empty queue;
  enqueue s → Q ;
  while Q not empty do
    dequeue v ← Q ;
    push v → S ;
    foreach neighbor w of v do
      // w found for the first time?
      if not visited[w] then
        equeue w → Q ;
      // shortest path to w via v ?
      if d[v] + d[w] < d[w] then
        append or update v → P[w] ;
        σ[w] ← σ[v] ;
```

```
            // found multiple shortest path to w ?
            elseif d[v] + d[w] = d[w] then
              append v → P[w];
              σ[w] ← σ[w] + σ[v];
            end
          end
        end
      end
      δ[v] ← 0, v ∈ V;
      // S returns vertices in order of non-increasing distance from s
      while S not empty do
        pop w ← S;
        for v ∈ P[w] do
          δ[v] ← δ[v] + (σ[v]/σ[w]) · (1 + δ[w]);
          if w ≠ s then
            C_eB[e] ← C_eB[e] + (σ[v]/σ[w]) · (1 + δ[w])
          end
        end
        if w ≠ s then C_vB[w] ← C_vB[w] + δ[w];
      end
    end
```

2.3 Parallel Implementation on the Cray XMT

The Cray XMT system is a scalable multithreaded high performance computing platform. The Cray XMT supports a global shared memory accessible by all processors on the system and can scale to a total of 8,192 processors. With a hardware multithreading mechanism to tolerate memory access latencies frequently encountered in irregular, graph-processing algorithms, the XMT machine is a suitable platform for parallel processing of large-scale power grid graphs.

In the proposed edge betweenness centrality computation, the modified Brandes' algorithm was called for each vertex in the power grid graph. In each iteration, all the shortest paths were identified between this given vertex and any other vertices in the graph, and then the algorithm accumulates the calculated edge betweenness values for particular edges. After all the vertices are scanned, edge betweenness of all edges will be available.

Each iteration is independent and can be executed in parallel since it analyzes independently-sourced shortest paths. We have used XMT-specific pragma **(#pragma mta parallel)** to guide the compiler in parallelizing other sections of the code. We have also used the atomic update pragma **(#pragma mta update)** before the corresponding source code statement for

$$C_{eB}[e] \leftarrow C_{eB}[e] + \frac{\sigma[v]}{\sigma[w]} \cdot (1 + \delta[w]);$$

By using **#pragma mta update,** the compiler knows that this statement is an update to the shared edge betweenness variable and that the update should be performed atomically.

3. Case Study

A power system model containing 760 buses (vertices), 977 branches (edges) was extracted from an area of the western US power grid. This model was then used to test and validate the proposed contingency selection method with the actual impact of power grid component failures based on performance indices [1].

GE Positive Sequence Load Flow Software (PSLF) [14] – a widely used, commercial grade power system analysis tool – was used to perform power grid contingency analysis. For each contingency result, a performance index [1] was calculated and all indices were ranked based on eq. 3:

$$PI = \sum_{i=1}^{N} \left(\frac{P_i}{P_{i\max}} \right)^2 \qquad \text{(eq. 3)}$$

where

N= the number of transmission branches (edges),

P_i = the power flow in each transmission branch (edge), and

$P_{i\max}$ = the power capacity for each transmission branch (edge).

In a power grid, we consider that, for each transmission branch, the more power carried, the more important this branch is. Therefore, the inverse of power flow at each branch, $1/P_i$, was used as the weight of the edges. To be consistent with the PI definition, the obtained edge betweenness was reweighted by the power capacity $P_{i\max}$ of the edge.

The distribution of edge betweenness for the 760-bus power system is shown in Figure 1. In Figure 1, we can see that there is a small portion of edges that have high edge betweenness scores, which means that this small portion of edges is important to this power grid graph. The same idea holds true in the real power system; there is a small set of transmission lines that are more critical compared with other lines. This is confirmed by the results for the full 14090-bus western US power grid (WECC) (see Figure 2).

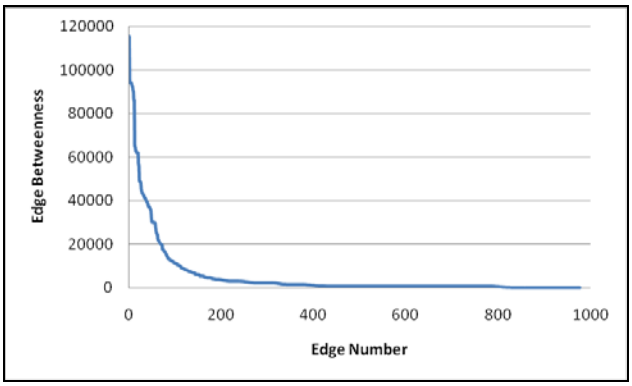

Figure 1: Edge betweenness distribution for the 760-bus system

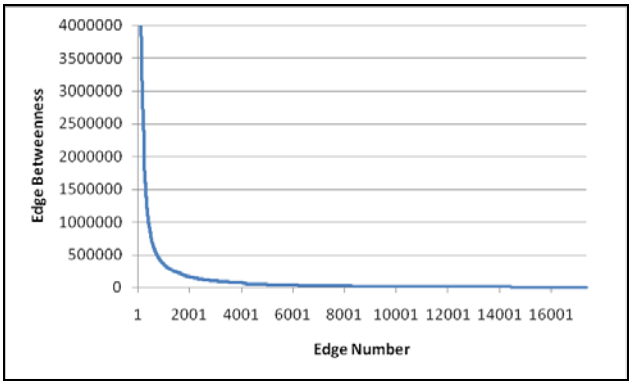

Figure 2: Edge betweenness distribution for the 14090-bus WECC system

The preliminary cross-validation results are shown in Figure 3, where the x-axis is the percentage of selected cases from contingency ranking results based on PIs, and the y-axis is the percentage of common cases between PI-based selection and edge-betweenness-based selection. From Figure 3, we can see that with 70% contingency cases selected (x-axis), the hit rate using edge betweenness can achieve 70% (y-axis). This high hit rate demonstrates the validity of the proposed edge-betweenness-based method.

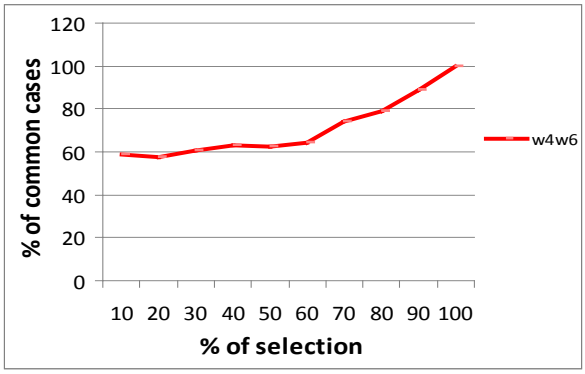

Figure 3: The cross-validation results for the 760-bus system

4. XMT Performance analysis

4.1 Study Cases

To analyze the XMT's computational performance for the application of edge betweenness centrality to power grid graphs, four power system models of different sizes were studied. The number of vertices and edges for the four graphs are shown in Table 1. The IEEE 118-bus system is a standard test system, available from [15]. The 760-bus system is the one used for validation in Section 3. The 46010-bus system was extended from the western US power grid (WECC), by adding more detailed power networks to represent some of the largest loads in the WECC system and matching power flow at points of connection.

Table 1: Study cases

Case name	Number of vertices	Number of edges
118-bus	118	179
760-bus	760	977
WECC	14090	17346
46010-bus	46010	57323

4.2 Resource Utilization

The Cray XMT machine used for this study has four compute blades with four multithreaded Threadstorm processors at 500 MHz. Each Threadstorm processor can have up to 128 hardware threads for a total of 128GB of global shared memory [16].

On the XMT, it is possible to accurately obtain the processor utilization by measuring the number of clock cycles in which instructions have been issued versus the total number of clock cycles spent in the computation. The processor utilization is a measure of the effectiveness of

memory access latency hiding. We have measured the memory usage and processor utilization for the 46010-bus case. These results are presented in Table 2.

Number of processors	Memory usage (GB)	Utilization rate (%)
1	85.8	43
2	87	42.5
4	88.2	39.5
8	92	37.1
16	97.6	34.5

Table 2: Memory usage and utilization rate versus the number of processors for the 46010-bus system

As can be seen from Table 2, the processor utilization is high in spite of the irregular nature of the graph analysis algorithm being executed. The multithreaded capabilities of the Threadstorm processor are quite effective in covering the latencies of the irregular memory accesses required to traverse the graph in order to compute the betweenness centrality scores.

4.3 Scalability

The four study cases listed in Table 1 were used to test the scalability of the edge betweenness implementation on the Cray XMT machine. The computer execution time is plotted against the number of processors in Figure 4, and the corresponding speed-up curve is shown in Figure 5.

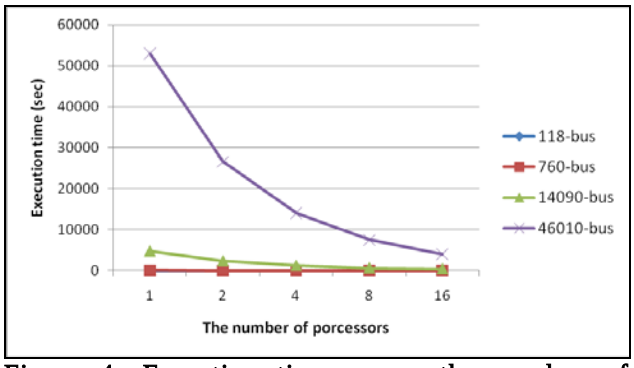

Figure 4: Execution time versus the number of processors for power graphs of various sizes on the Cray XMT

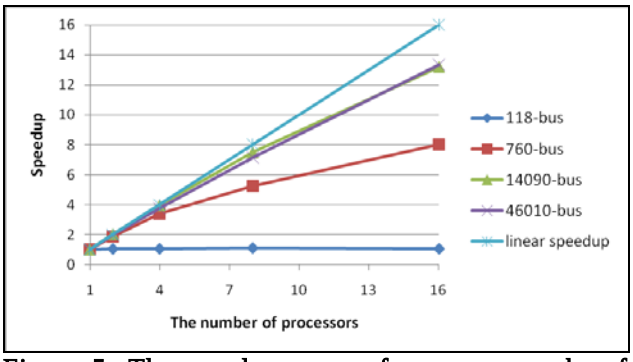

Figure 5: The speed-up curve for power graphs of various sizes on the Cray XMT

In Figure 4 and 5, it is clear that the scalability improves for larger problem sizes. For large graphs, the speed-up is close to linear scalability. This indicates that the Cray XMT is very suitable for the proposed betweenness centrality analysis method operating on large power grid graphs.

5. Comparison between the Cray XMT and a shared-memory machine

To compare the performance between the Cray XMT machine and an equivalent scalable shared-memory machine, we used a Hewlett-Packard SuperDome machine, located at PNNL. The SuperDome runs a version of the betweenness centrality application written using the OpenMP share-memory programming model.

The configuration of the SuperDome machine is as follows:

- 64 dual-core 1.6GHZ Itanium processors (128 cores)
- 24 MB of cache per processor
- 256 GB RAM
- 12.6 TB of disk space.

The same four cases in Table 1 were used to compare the performance of the Cray XMT and the SuperDome machine. In order to conduct a reasonably fair comparison, the same amount of memory (128 GB) was reserved for each run. Figure 6 and Figure 7 show the plots of computer execution time and the speed-up numbers, respectively.

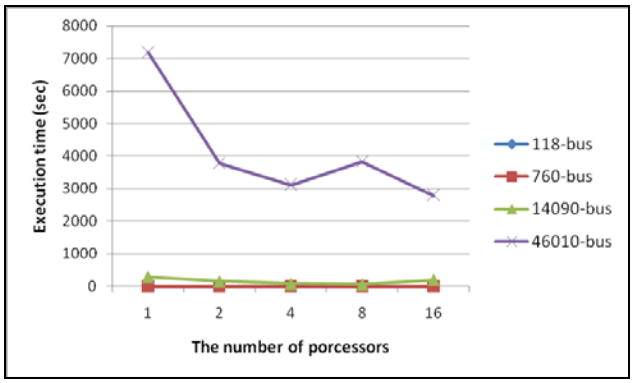

Figure 6 : Execution time versus the number of processors for power graphs of various sizes on the SuperDome

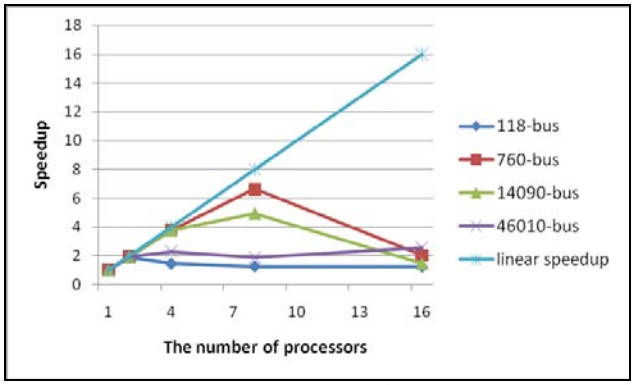

Figure 7: The speed-up curve for power graphs of various sizes on the SuperDome

Comparison of Figure 5 and Figure 7 indicates that the Cray XMT machine achieves superior scalability for graph edge betweenness computation over the SuperDome machine. Figure 5 also shows a very consistent performance of the XMT machine as the graph size increases, while the SuperDome machine exhibits somewhat unpredictable performance, as shown in Figure 7. This inconsistency in the SuperDome performance can be mainly attributed to the variability in the use of its local caches. Figure 6 shows that for the 46010-bus system, the execution time with eight processors is even longer than that with four processors. We believe that the reason for this is potential false sharing between the processors' caches, which increases as the number of processors grows. The cache in the SuperDome machine is local to individual computer nodes. For the case with eight processors, two compute nodes are used. Processors in one node need to access data in the cache attached to other node. Therefore, more overhead is introduced by inter-node communication, which degrades the overall computational performance. As a result, it takes longer to execute the code on eight processors than on four processors. When 16 processors (four compute nodes) are used, the execution time improves again, but it is not much better than on four processors, in spite of the work per processor having been reduced by a factor of four. Further analysis is required in order to better understand and characterize the behavior of this application on a cache-coherent platform (like SuperDome).

Table 3 shows the execution time and the ratio of execution time ($T_{XMT}/T_{SuperDome}$) with 16 processors for the four power grid graphs on the Cray XMT. For these four cases, we can see that the execution time on the SuperDome is less than that on the Cray XMT. One factor contributing this result is the difference in the processor clock speed. There is a factor of 3.2, comparing SuperDome's 1.6 GHz and XMT's Threadstorm 500 MHz. The relative ratio of execution time considering the factor of 3.2 is also shown in Table 3. The relative ratio is equivalent to a comparison of the computational efficiency if both SuperDome and XMT have the same clock speed. According to the relative ratios, Table 3 shows that the XMT machine would have already outperformed the SuperDome machine for the 14090-bus WECC power system as the ratio 0.58 is well below 1.0.

But even without considering the clock speed difference, the ratio of execution time ($T_{XMT}/T_{SuperDome}$) is approaching 1.0 when the size of power grid graphs increases. Another factor for SuperDome's faster execution is the cache effect. Especially for smaller graphs, the SuperDome can take advantage of its large caches, in which most or even all data of smaller graphs can fit, and thus the average memory access time is reduced dramatically. When the graph is so large that the data can not fit into the caches and swapping between caches and the main memory is necessary, the memory access time increases since most of the data won't be available in the caches. The advantage of caches diminishes, and thus the overall computational time increases prominently. With the multithreaded mechanism and excellent scalability, we expect that the current Cray XMT configuration, though with slower processors, would outperform the SuperDome in terms of execution time when the graph size is larger than a threshold.

Table 3: Execution time for four study cases on the SuperDome and the Cray XMT

Case name	$T_{SuperDome}$ (sec)	T_{XMT} (sec)	Ratio (=$T_{XMT}/T_{SuperDome}$)	Relative ratio (=Ratio/3.2)
118-bus	0.013	0.531	40.91	12.78
760-bus	0.336	1.681	5.00	1.56
WECC	196.79	364.64	1.85	0.58
46010-bus	2791.40	3969.30	1.42	0.44

To estimate the graph size threshold, a preliminary extrapolation is performed on the data shown in Table 3.

Figure 8 shows the result as well as the identified trending function. Based on the trending function, we can estimate that when the graph size is larger than that of 53,000 vertices, the Cray XMT would take less time to compute edge betweenness on 16 processors than the SuperDome. The testing is ongoing with a 60,000-bus power system case to validate the estimated threshold and to further demonstrate the computational performance of the XMT machine.

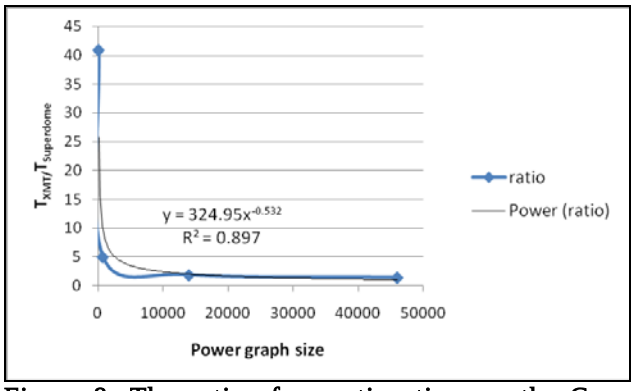

Figure 8: The ratio of execution time on the Cray XMT and the SuperDome versus the size of power grid graph

Conclusion and Future Work

A contingency selection method is developed by applying edge betweenness centrality in graph theory to power grid topology. This selection method can identify low-impact components whose failure is of little importance to power grid stability. Removing them from analysis reduces the combinatorial number of contingency cases. Cross-validation of the proposed method has been conducted. This method has been implemented on the Cray XMT machine, taking the advantage of the graph processing capability of Cray XMT's Threadstorm nodes and its programming features. The test results show the excellent scalability of Cray XMT and better performance than a shared-memory machine for large graphs.

In the future, we will take advantage of the Cray XMT's hybrid architecture of Threadstorm and Opteron nodes. Not only are Threadstorm nodes used to perform contingency selection, but Opteron nodes are used to perform the floating point computation of actual contingency analysis. Further work will focus on the communication between Threadstorm nodes and Opteron nodes.

Acknowledgments

This work is supported by the Center for Adaptive Supercomputing Software – Multi-Threaded Architectures (CASS-MT) funded by the Department of Defense and by the Electricity Infrastructure Operations Initiative of the Pacific Northwest National Laboratory. The Pacific Northwest National Laboratory is operated by Battelle for the U.S. Department of Energy under Contract DE-AC06-76RL01830. Acknowledgment is extended to Jarek Nieplocha, Pak C. Wong, and Ross Guttromson, all with the Pacific Northwest National Laboratory, for productive discussions and suggestions.

References

[1] G.C. Ejebe and B. F. Wollenberg, "Automatic Contingency Selection," *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-98, No. 1, pp. 92-104, Jan./Feb. 1979.

[2] T.A. Mikolinnas and B. F. Wollenberg, "An Advanced Contingency Selection Algorithm," *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-100, No. 2, pp. 608-617, Feb. 1981.

[3] J. Zaborszky, F.W. Whang and Prasad, "Fast Contingency Evaluation Using Concentric Relaxation," *IEEE Trans. on Power Appa-atus and Systems*, vol. PAS-99, No. 1, pp. 28-36, Jan./Feb. 1980.

[4] W. F. Tinney, V. Brandwajn and S. M. Chan, "Sparse Vector Method," *IEEE Trans. on Power Apparatus and Svstems*, vol. PAS-104, No. 2,pp. 295-301, Feb. 1985.

[5] S. M. Chan and V. Brandwajn, "Partial Matrix Refactorization," *IEEE Trans. on Power Systems*, vol. PWRS-l, No. 1, pp. 193-200, Feb. 1986.

[6] V. Brandwajn and M. G. Lauby, "Complete Bounding for AC Contingency Analysis," *IEEE Trans. On Power Systems*, vol. PWRS-4, No. 2, pp. 724-729, May 1990.

[7] A. P. Sakis Meliopoulos and C. Cheng, "A Hybrid Contingency Selection method," in *Proceedings of the 10th Power System Computation Conference*, Graz, Austria, Aug. 1990, pp. 605-612.

[8] Sun Wook Kang, A. P. Meliopoulos, "Contingency Selection via Quadratized Power Flow Sensitivity Analysis," in *Proceedings of the IEEE 2002 Power Engineering Society Summer Meeting*, vol.3, pp.1494-1499.

[9] Freeman, L. C., "A set of measures of centrality based on betweenness", *Sociometry*, 1977, 40:35-41.

[10] Anthonisse, J. M. "The rush in a directed graph", Technical Report BN 9/71, Stichting Mathematisch Centrum, Amsterdam, 1971.

[11] "Clustering Using Betweenness Centrality", http://iv.slis.indiana.edu/sw/bc.html

[12] Ulrik Brandes, "A Faster Algorithm for Betweenness Centrality", *Journal of Mathematical Sociology*, Vol. 25, 2001, pp. 163-177.

[13] E. W. Dijkstra, "A note on Two Problems

Inconnexion with Graphs", *Numerische Mathematik*, 1 (1959), S. 269–271.
[14] http://www.ge-energy.com/prod_serv/products/utility_software/en/ ge_pslf/index.htm
[15] http://www.ee.washington.edu/research/pstca/pf118/pg_tca118bus.htm
[16] John Feo, D. Harper, S. Kahan, and P. Konecny, "ELAORADO", *CF'05*, May 4-6, 2005, Ischia, Italy, 2005 ACM

About the Authors

Yousu Chen is a Research Engineer at the Pacific Northwest National Laboratory in Richland Washington. He is an IEEE member and the Chair of the Richland Chapter of the Power & Energy Society. He can be reached at 902 Battelle Blvd. MSIN K1-85, Richland WA, 99352, E-Mail: yousu.chen@pnl.gov.

Shuangshuang Jin is a Research Engineer at the Pacific Northwest National Laboratory in Richland Washington Her main areas of interest are high performance computation and visualization. Dr. Jin can be reached at 902 Battelle Blvd. MSIN K1-85, Richland WA, 99352, E-Mail: shuangshuan.jin@pnl.gov.

Daniel Chavarría-Miranda is a Senior Research Scientist at the Pacific Northwest National Laboratory in Richland Washington. He can be reached at 902 Battelle Blvd. MSIN K7-90, Richland WA, 99352, E-Mail: daniel.chavarria@pnl.gov.

Zhenyu (Henry) Huang is currently a Senior Research Engineer at the Pacific Northwest National Laboratory, Richland, WA. His research interests include power system stability and control, high-performance computing applications, and power system signal processing. Dr. Huang can be reached at zhenyu.huang@pnl.gov .