High Performance Computing for Contingency Analysis of Power Systems

Anju Raj V
Department of Electrical and Electronics, FISAT, Kerala, India

Jino M Pattery
Senior Engineer Kalkitech

Surumi Hassainar
Department of Electrical and Electronics, FISAT, Kerala, India

Abstract

Preventing power black-outs in the electrical power grid is a complex task. This emphasizes the need for contingency analysis, which involves understanding and mitigating potential failures in power grid elements such as transmission lines. When considering the potential for multiple simultaneous failures (known as the N-x contingency problem), contingency analysis becomes a massively computational task. In this paper we describe a novel hybrid computational approach to contingency analysis in a power system. This approach exploits the conventional massively parallel computer cluster to identify likely simultaneous failures that can lead to widespread cascading power failures that have massive economic and social impact on society. When deployed in power grid operations, it will increase the grid operator's ability to deal effectively with outages and failures with power grid components while preserving stable and safe operation of the grid. The paper describes the architecture of our solution and presents preliminary performance results that validate the efficacy of our approach.

1. Introduction

In a power grid, multiple redundant lines between nodes in the network are provided so that power can be routed using a variety of paths from any power plant to any load center. The exact route chosen is based on the economics of the transmission path and the cost of power. A problem in the electric power grid, such as a transmission line going out of service due to contact with vegetation, can lead to various issues. So if large amounts of electric power are being transferred from one geographic area to another, the loss the connecting line will have impacts on loads and voltages across the grid. Loads may be lost and voltages may drop or even collapse in various areas, leading to power outages.

These unexpected outages are referred to as power system "contingencies" and the grid operators manage the system by ensuring that any single contingency will not propagate into a cascading blackout. This is referred to as the N-1 contingency standard issued by the North American Electric Reliability Corporation (NERC). If multiple contingencies occur simultaneously, the grid has to be restored urgently to a normal condition. This indicates the need for N-x contingency analysis, i.e. analysis of the potential simultaneous occurrence of multiple contingencies. Such analysis can prepare grid operators with mitigation procedures so as to avoid cascading failures. N-x contingency analysis is an extremely complex computational task.

In this paper, the computational approach for N-x contingency analysis is described. This approach exploits the multithreaded architecture to select likely contingencies from the power grid network. The identified contingencies are transferred to a conventional compute cluster to perform contingency analysis on selected cases based AC power flow. This analysis generates potentially terabytes of data which has to be transferred back to identify likely power grid vulnerabilities and utilize advanced visual analytical methods to present the results to operators for remedial actions.

In this project the potential of high performance computing for contingency analysis is dealt with. The framework of N-1 contingency analysis is established and computational load balancing schemes are studied and also implemented with high performance LINUX clusters with Parallel Virtual Machine (PVM) as the programming environment. Each contingency case gets dynamically allocated to individual processors with distributed memory architecture which is well suited for applications with minimum data communication requirements.

2. Contingency Analysis

Many wide spread blackouts have occurred in the past in interconnected power systems and hence it is necessary to ensure that power system should be operated most economically so that power is delivered reliably. Contingency analysis is a well known function in modern Energy Management Systems (EMS). The goal of this power system analysis function is to give the operator information about the static security. Contingency Analysis of a power system is a major activity in power system planning and operation. In general an outage of one transmission line or transformer may lead to over loads in other branches and/or sudden system voltage rise or drop. Contingency analysis is used to calculate such violations. Contingency analysis is performed to shortlist a specified set of contingencies from a list of contingencies and to rank them according to their severity.

In a power utility control center, contingency analysis (CA) is one of the "security analysis" applications performed that differentiates an Energy Management System (EMS) from a less complex SCADA system. Its purpose is to analyze the power system in order to identify the overloads and problems that can occur due to a "contingency". Contingency is an abnormal condition in electrical network. It put whole system or a part of the system under stress. It occurs due to sudden opening of a transmission line, generator tripping, sudden change in generation or sudden change in load value. Contingency analysis provides tools for managing, creating, analyzing, and reporting lists of contingencies and associated violations.CA is used as a study tool for the off-line analysis of contingency events, and as an on-line tool to show operators what would be the effects of future outages.

Security is determined by the ability of the system to withstand equipment failure. Weak elements are those that present overloads in the contingency conditions (congestion). Standard approach is to perform a single (N-1) contingency analysis simulation. CA is therefore a primary tool used for preparation of the annual maintenance plan and the corresponding outage schedule for the power system.

Although computer technology has made almost all aspects in human activities to run smoothly some events are unpredictable and beyond our control. In power system operation unpredictable events are termed as contingencies. Contingencies can be divided into generator outages, load outages and line outages etc.

The selection of contingencies is needed to reduce the computation time of contingency analysis. Among the various contingencies only the line outages are considered in this project since they are the most common contingencies and for simplicity single line outages are considered .in contingency analysis a load flow analysis is done for each contingency and L-indices for load buses are calculated in each case. Severity index is calculated for each contingency based on a composite criterion. The contingency ranking is done using various criteria, such as bus voltage profiles and voltage stability indexes of load buses are carried out. The ranking is evaluated using the composite criteria.

3. Hybrid Computing for Contingency Analysis

Two major types of HPC architectures are available that are built with the multi-core/many-core processors, shared-memory architecture and distributed-memory architecture. When applying HPC technologies, a key success factor is the match of computer architectures with problem characteristics. The shared memory architecture has the main memory block commonly accessible by all the processors in a random non-uniform manner. Shared memory architecture is useful for efficient implementation of sparse matrix and irregular computations such as power system state estimation. The distributed memory architecture consists of processors with local memory. High-speed data links are used between processors for communication. This architecture does not have the issue with main memory access, but inter-processor communication can be a bottleneck if an application requires frequent data exchange between processors. The distributed memory architecture well suits applications which can be divided into sub-tasks with minimum data communication requirements. Power system contingency analysis is one of such problems.

Given the sheer number of contingency cases in the problem space and the real-time requirements for power grid operations, today’s industry algorithms and tools are not able to handle comprehensive contingency analysis as described in the previous section. A comprehensive contingency analysis process includes the following three elements:

1) contingency selection.

2) parallel contingency analysis.

3) post-processing of contingency analysis.

4. Need for High Performance Contingency Analysis

Contingency analysis is an essential part of power grid and market operations. Traditionally, contingency analysis is limited to be selected "N-1" cases within a balancing authority's boundary. Power grid operators manage the system in a way that ensures any single credible contingency will not propagate into a cascading blackout, which approximately summarizes the "N-1" contingency standard established by the North American Electric Reliability Corporation (NERC).

Though it has been a common industry practice, analysis based on limited "N-1" cases may not be adequate to assess the vulnerability of today's power grids due to new development in power grid and market operations.

As for power grid operation, recent cascading failures reveal the need of "N-x" contingency analysis. The old assumption is that a cascading failure is caused by a single credible contingency. However, multiple unrelated events may occur in a system and result in cascading failures. Therefore, "N-2" and even higher order ("N-x") contingency events need to be analyzed. "N-x" contingency analysis or even just more comprehensive "N-1" analysis is very challenging due to the combinatory number of contingencies and the extremely large amount of computational time.

Obviously, high performance computing application is a must for meeting the need of massive power system contingency analysis. The performance of high-performance computing application for contingency analysis heavily relies on computational load balancing. A well-designed computational load balancing scheme considering the CPU speed, network bandwidth and data exchange latency is key to the success.

5. Parallel Contingency Analysis

Contingency analysis is naturally a parallel process because multiple contingency cases can be easily divided onto multiple processors and communication between different processors is very minimal. Therefore, cluster-based parallel machines are well suited for contingency analysis. For the same reason, the challenge in parallel contingency analysis is not on the low-level algorithm parallelization but on the computational load balancing (task partitioning) to achieve the evenness of execution time for multiple processors.

The framework of parallel contingency analysis is shown in Figure 1. Each contingency case is essentially a power flow run. In our investigation, full Newton-Raphson power flow solution is implemented. Given a solved base case, each contingency updates its admittance matrix with an incremental change from the base case. One processor is designated as the master process (Proc 0 in Figure 4) to manage case allocation and load balancing, in addition to running contingency cases.

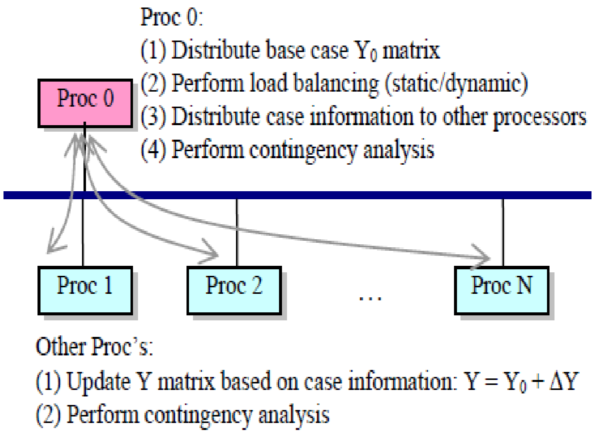
Proc 0:
(1) Distribute base case Y_0 matrix
(2) Perform load balancing (static/dynamic)
(3) Distribute case information to other processors
(4) Perform contingency analysis
Proc 0
Proc 1
Proc 2
…
Proc N
Other Proc's:
(1) Update Y matrix based on case information: $Y = Y_0 + \Delta Y$
(2) Perform contingency analysis

Fig. 1: Framework of Parallel Contingency Analysis

6. Computational Load Balancing Schemes

The straightforward load balancing of parallel contingency analysis is to pre-allocate equal number of cases to each processor, i.e. static load balancing. The master processor only needs to allocate the cases once at the beginning. Due to different convergence performance for different cases, the power flow run may require different number of iterations and thus take different time to finish. The extreme case would be non-converged cases which iterate until the maximum number of iterations is reached. The variations in execution time result in unevenness, and the overall computational efficiency is determined by the longest execution time of individual processors. Computational power is not fully utilized as many processors are idle while waiting for the last one to finish.

Another load balancing scheme is to allocate tasks to processors based on the availability of a processor, i.e. dynamic load balancing. In another word, the contingency cases are dynamically allocated to the individual processors so that the cases are more evenly distributed in terms of execution time by significantly reducing processor idle time. The scheme is based on a shared task counter updated by atomic fetch-and-add operations. The master processor (Proc 0) does not distribute all the cases at the beginning. Instead, it maintains a task counter. Whenever a processor finishes its assigned case, the processor requests more tasks from the master processor and the task counter is updated. This process is illustrated in Figure 3. Different from the evenly-distributed number of cases on each processor with the static scheme, the number of cases on each processor with the dynamic scheme may not be equal, but the computation time on each processor is optimally equalized.

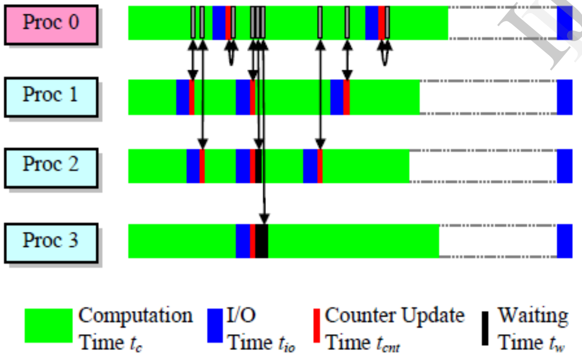

Fig. 2: Task- counter- based dynamic load balancing scheme

7. Dynamic Load Balancing Scheme

The dynamic computational load balancing scheme balances execution time among processors better than the static scheme. But the cost is the overhead of managing the task counter.

As shown in Figure 2, the execution time of each case consists of four parts: t_c – the computation time spent on solving one contingency case, tio – the I/O time used to write the results to disks, t_{cnt} – the time to update the task counter, and t_w – the time to wait for the master processor to respond with a new case assignment when counter congestion occurs. Running all the cases on only one processor would take a total time as estimated in (1):

$$t_{total} = \sum_{i-1}^{N_c} (t_c^{(i)} + t_{i0}^{i}) = N_C (t_c^{-} + t_{i0}^{-}) \qquad (1)$$

where N_C is the total number of cases, and t_c and t_{io} are the average computation time and I/O time, respectively. On one processor, there is no counter management needed, so no t_{cnt} and t_w should be included in (1).

Running the cases on multiple processors with dynamic load balancing scheme would evenly distribute the total time in (1), but involves counter management. If the total number of processor is N_P, the worst-case scenario with counter congestion is that all the N_P counter updates arrive at the same time at the master processor. Then the first processor has not waiting time, the second waits for time t_w, and the last one has the longest waiting time given by $(N_P -1)t_w$. The average waiting time of a processor can be estimated as:

$$t_{w,N} \approx \frac{\frac{N_C}{N_P} \sum_{i=1}^{N_p} (i-1) t_{w-}}{N_P} = \frac{N_C}{N_P} \frac{(N_p - 1) t_w}{2} \qquad (2)$$

Therefore, the total wall clock time required to run all the contingency cases can be estimated as (3):

$$t_{total,N_P} \approx \frac{\sum_{i=1}^{N_C} (t_c^i + t_{i0}^i)}{N_P} + \frac{N_C}{N_P} t_{cnt} + \frac{N_C}{N_P} \frac{(N_p - 1) t_w}{2}$$

$$= \frac{N_C}{N_P} \left[t_c^{-} + t_{i0}^{-} + t_{cnt} + \frac{(N_p - 1) t_w}{2} \right] \qquad (3)$$

The speedup performance of dynamic load balancing scheme can be expressed as the following conservative estimate:

$$S_{N_P} = \frac{t_{total}}{t_{total,N_P}}$$

$$= \frac{N_C(t_c^{-} + t_{i0}^{-})}{\frac{N_C}{N_P}\left[t_c^{-} + t_{i0}^{-} + t_{cnt} + \frac{(N_P\ -1)t_w}{2}\right]} \qquad (4)$$

$$= N_P \frac{(t_c^{-} + t_{i0}^{-})}{t_c^{-} + t_{i0}^{-} + t_{cnt} + \frac{(N_P\ -1)t_w}{2}}$$

Several observations can be drawn from (4):

1) It is clearly shown that the dynamic load balancing scheme is scalable with the number of cases as the speedup performance is irrelevant to the number of cases, N_C.

2) If the counter update is instantaneous and no counter congestion would occur, i.e. $t_{cnt} = 0$ and $t_w = 0$, then the ideal speedup performance would be N_P, equal to the number of processors.

3) For practical implementation, improving speedup performance would require to minimize the overhead t_{cnt} and t_w.

4) Counter update time t_{cnt} is mainly determined by the network bandwidth and speed. Minimizing t_{cnt} usually means to choose high-performance network connection between processors.

5) Waiting time t_w is due to counter congestion. Though more processors would improve the speedup, but they also increase the possibility of counter congestion as shown in (4).

8. Parallel Virtual Machine

Parallel Virtual Machine (PVM) is a tool for parallel networking of computers. It is designed to allow a network of heterogeneous Linux and/or Windows machines to be used as a single distributed parallel processor. Thus large computational problems can be solved more cost effectively by using the aggregate power and memory of many computers. The software is very portable and the source code is available free through netlib. PVM enables users to exploit their existing computer hardware to solve much larger problems at less additional cost. PVM was a step towards modern trends in distributed processing and grid computing.PVM is free software, released under both the BSD License and the GNU General Public License.

9. Communication Between Tasks

Once new tasks are spawned and programs are compiled, actual parallel programming can be done. The different tasks can communicate with each other. In PVM, task-to-task communication is done with message passing. The figure.3 below shows the communication processes in a parallel virtual machine architecture. Contingency analysis is performed on three test cases: IEEE 14 bus system, IEEE 30 bus system and IEEE 57 bus systems.

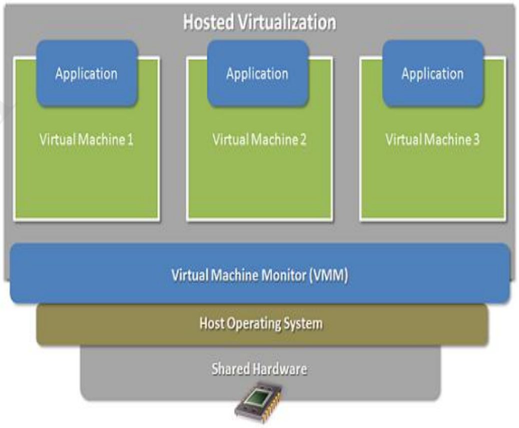

Fig.3: Communication processes in PVM architecture

10. Test Results

10.1. Contingency Analysis test results:

The contingency analysis was done in scilab using Newton Raphson power flow method for line contingencies in the IEEE 14 bus, IEEE 30 bus and IEEE57 bus systems and the results obtained are shown in tabe below.

TABLE I
LOAD FLOW RESULTS

Iteration No	Time taken (in sec) (14 bus) [20 cases]	Time taken (in sec) (30 bus) [41 cases]	Time taken (in sec) (57 bus) [80 cases]
1	6.816000	28.591000	345.724000
2	6.791000	28.610000	346.874000
3	6.777000	28.590000	346.145000
4	6.847000	28.610000	349.751000
5	6.794000	28.608000	347.933000
6	6.736000	28.696000	350.139000
7	6.838000	28.740000	349.273000
8	6.862000	28.816000	349.485000
9	6.861000	28.693000	348.307000
10	6.838000	28.796000	344.497000

11. Conclusion

The computational performance of the dynamic load balancing scheme is analyzed, and the results provide guidance in using high-performance computing machines for large number of relatively independent computational jobs such as power system contingency analysis. The test results indicate excellent scalability of the dynamic load balancing scheme.

12. References

[1] U.S.-Canada Power System Outage Task Force, Final Report on the August 14, 2003 Blackout in the United State and Canada: Causes and Recommendations, April 2004. At https://reports.energy.gov/.

[2] J. Deuse, K. Karoui, A. Bihain, J. Dubois, Comprehensive approach of power system contingency analysis, 2003 IEEE Power Tech Conference Proceedings, Bologna, Volume 3, 23-26 June, 2003.

[3] R.H. Chen, G. Jingde, O.P. Malik, W. Shi-Ying,N. Xiang, Automatic contingency analysis and classification, The Fourth International Conference on Power System Control and Management, 16-18 April, 1996.

[4] Y. Chen, S Jin, D Chavarría-Miranda, Z Huang,Application of Cray XMT for Power Grid Contingency Selection, Proceedings of Cray User Group 2009, Atlanta, GA, May 4-7, 2009.

[5] Z. Huang, Y. Chen, J. Nieplocha, Massive Contingency Analysis with High Performance Computing, Proc. IEEE PES General Meeting, Calgary, Canada, July 2009

[6] Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar. 2003), 1289-1305.

[7] D. N. Kosterev, C. W. Taylor, and W. A. Mittelstadt, “Model Validation for the August 10, 1996 WSCC System Outage,” *IEEE Trans. Power Syst.*, vol. 14, no. 3, pp. 967-979, August 1999.

[8] NERC standards, Transmission System Standards – Normal and Emergency Conditions, available at www.nerc.com

[9] Quirino Morante, Nadia Ranaldo, Alfredo Vaccaro, and Eugenio Zimeo, "Pervasive Grid for Large-Scale Power Systems Contingency Analysis," IEEE Transactions on Industrial Informatics, vol. 2, no. 3, August 2006

[10] Chen, R.H.; Jingde Gao; Malik, O.P.; Shi-Ying Wang; Nian-De Xiang; "Automatic contingency analysis and classification," The Fourth International Conference on Power System Control and Management, 16-18 April, 1996.