Multicore Methods to Accelerate Ship Power System Simulations

Fabian M. Uriarte
Center for Electromechanics
The University of Texas at Austin
Austin, TX, USA
f.uriarte@cem.utexas.edu

Christian Dufour
Opal-RT Technologies
Montréal, Québec
Canada
christian.dufour@opal-rt.com

***Abstract*—Two methods to partition and parallelize the simulation of large-scale shipboard power systems on multicore computes are demonstrated. The first method is node tearing, which is used for offline simulation. The second is the state-space nodal method, which is used for real-time simulation. Both methods are benchmarked against *MATLAB*/*Simulink* 2012b for speed and accuracy The simulation model is a notional shipboard power system having characteristics of AC-radial, 450 V, 60 Hz, three-phase, delta-ungrounded power system. The parallel simulation results show speedups in excess of one order of magnitude and general agreement in accuracy.**

I. INTRODUCTION

Experience [1] in simulation shows that the run time of large-scale shipboard power system simulations are too time consuming to be useful in early-stage design trades. These lengthy run times limit the rate at which case studies can be run and consume significant research resources.

The Center for Electromechanics (CEM) at the University of Texas at Austin (UT) and Opal-RT Technologies, Inc. are developing partitioning methods to parallelize the simulation of large power system models on multicore processors. Proper exploitation of multicore technology, as such, requires the re-design of existing simulation methods to embrace parallelism. This paper demonstrates two simulation methods to parallelize power system simulations on shared-memory multicore processors. The first method is developed by CEM, and is implemented in its solver known as *CEMSolver* [2, 3]. The second method is called the State-Space Solver, developed by Opal-RT Technologies, and is implemented in the ARTEMiS solver suite [4, 5].

II. SHIPBOARD POWER SYSTEM MODEL

The shipboard power system model used in this work is shown in Figure 7. This model has characteristics of existing Navy shipboard power systems, but does not represent any particular system. The information to build this model was taken from [6-18], and was used to build a computer model in *MATLAB/Simulink* 2012b [19] and *SimPowerSystems*.

The model represents the electric distribution (ship service) side of an electromechanical shipboard, and it is *not* related to an integrated power system or all-electric ship [20]. The power apparatus (i.e., equipment) appearing in Figure 7 was modeled as follows. The generators were modeled as three-phase voltage sources (450 V, 60 Hz) behind sub-transient impedances [13, 14, 16, 18]. Cables were modeled as ungrounded π-sections (*LSTSGU* three-conductor, shipboard power cable, 450 V, three-phase, 60 Hz [8, 9]). The transformers were modeled as 25-kVA single-phase units (450:120 V) connected in delta on both sides [11]. The loads were modeled as static impedance loads with the consumptions tabulated in TABLE I. The columns in TABLE I. indicate the volt-ampere rating in kVA, power consumption in kW, reactive power draw or injection in kVar, the power factor (PF), and the full-load ampere (FLA) draw. The protective devices were modeled as time-varying resistors. Each fault was modeled as three-phase breaker connected in an ungrounded-wye configuration to produce a-b-c faults. The breakers close-in and begin their opening actions at the indicated times; however, the current in each phase is *not* interrupted until the next available current zero-crossing in each phase.

III. MULTICORE PARTITIONING METHODS

The model in Figure 7 presents a large number of nodes, power apparatus, and equation count (885 independent state-variables, and 1,188 inductors and capacitors in total). The two partitioning methods demonstrate herein will show that the execution time of this model can be accelerated with two different targets: a *Windows* desktop and a *Linux*-based multicore machine. A high-level summary of these two methods follows.

A. Node Tearing (offline)

The first partitioning method is *node tearing* [2, 3], and it is used to accelerate (offline) simulations using an everyday *Windows*-based multicore computer. This is accomplished by, first, partitioning the power system model, and then, parallelizing its solution.

Consider the arbitrary three-phase bus shown in Figure 1. This bus represents an *arbitrary* three-phase node in the shipboard power system model shown in Figure 7. (Identification of boundary three-phase nodes can be achieved using graph theory [21].) Assuming it is desirable to create

This work is sponsored by the Office of Naval Research as part of the Electric Ship Research and Development Consortium.

p=3 partitions (or subsystems) at this three-phase node, insertion of unknown-valued current sources at this boundary results in the in-line current sources shown in Figure 2. From circuit theory, it is possible to tear each current source as shown in Figure 3, which results in a partitioned network with *p*=3 subsystems. Tearing current sources as such results in a set of doubly-bordered block-diagonal matrix equations for the power system network. These types of equations can be solved with "fork/join" algorithms [22, 23] on a multicore computer, which results in pronounced speedups in large models.

The mathematics behind this tearing principle can be found in [24, 25], which is rooted in diakoptics [26, 27]. Readers may notice, from Figure 3, that the current sinks in subsystem 1 can be expressed as the sum of the current injections in subsystems 2 and 3. This dependency must be observed during software implementation.

TABLE I. LOAD POWER RATINGS

L01	333.33	300.00	145.30	0.90	428.18
L02	6.60	6.60	-	1.00	8.48
L03	6.60	6.60	-	1.00	8.48
L04	11.77	10.00	6.20	0.85	15.11
L05	35.29	30.00	18.59	0.85	45.33
L06	35.29	30.00	18.59	0.85	45.33
L07	38.11	36.20	11.90	0.95	48.95
L08	38.11	36.20	11.90	0.95	48.95
L09	38.11	36.20	11.90	0.95	48.95
L10	70.00	42.00	56.00	0.60	89.92
L11	190.53	181.00	59.49	0.95	244.73
L12	315.79	300.00	98.61	0.95	405.64
L13	6.60	6.60	-	1.00	8.48
M01	4.55	4.55	-	1.00	5.84
M02	4.55	4.55	-	1.00	5.84
M03	42.87	42.87	-	1.00	55.07
M04	52.05	52.05	-	1.00	66.85
M05	52.05	52.05	-	1.00	66.85
M06	52.05	52.05	-	1.00	66.85
M07	52.05	52.05	-	0.86	66.85
M08	86.74	86.74	-	1.00	111.42
M09	86.74	86.74	-	1.00	111.42
M10	134.82	134.82	-	1.00	173.18
M11	134.82	134.82	-	1.00	173.18
M12	134.82	134.82	-	1.00	173.18
M13	134.82	134.82	-	1.00	173.18
M14	134.82	134.82	-	1.00	173.18
M15	134.82	134.82	-	1.00	173.18
M16	207.22	207.22	-	1.00	266.18
M17	207.22	207.22	-	1.00	266.18
M18	207.22	207.22	-	1.00	266.18
M19	207.22	207.22	-	1.00	266.18
T01	75.00	70.00	26.93	0.93	89.92
T02	75.00	70.00	26.93	0.93	89.92
T03	75.00	70.00	26.93	0.93	89.92
T04	75.00	70.00	26.93	0.93	89.92
T05	75.00	70.00	26.93	0.93	89.92
T06	75.00	70.00	26.93	0.93	89.92
T07	75.00	70.00	26.93	0.93	89.92
T08	75.00	70.00	26.93	0.93	89.92
T09	75.00	70.00	26.93	0.93	89.92
T10	75.00	70.00	26.93	0.93	89.92
T11	75.00	70.00	26.93	0.93	89.92
Total:	4,023	3,863	438		4,107

*Loads of type "T" have a leading power factor

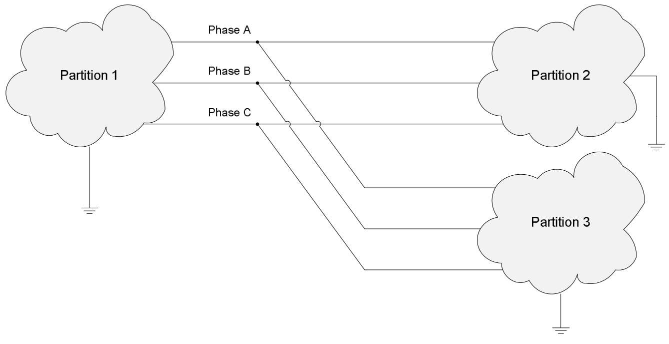

Figure 1 Tearing a bus to produce three subsystems

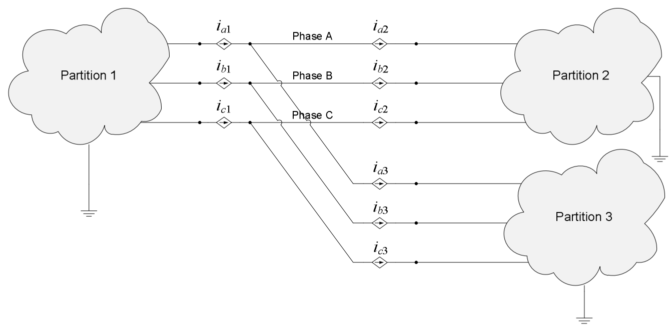

Figure 2 Addition of unknown-valued current sources at disconnection point

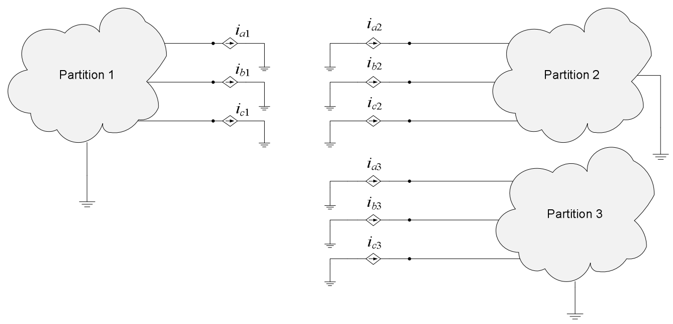

Figure 3 Creation of three subsystems from a bus disconnection point creates *six* boundary variables instead of nine.

B. State-space Nodal

The second multicore partitioning method is the *state-space nodal* (SSN) method [4, 5]. The SSN solver is also a node-tearing method, in that system nodes are torn apart to produce power system partitions. However, the SSN uses voltage sources at the subsystem boundaries instead of current sources.

In contrast to the first node tearing method used for *offline* simulations, the SSN method is used primarily to optimize *real-time* simulations of *SimPowerSystems* models. In SSN, the user manually defines the partition boundaries using special blocks to represent SSN (interface or boundary) nodes. These nodes become nodes with implicit voltage and current relations at the boundary of all partitions. Within each partition, state-space equations are formed to solve the internal state of each subsystem. The simultaneous solution of all partitions is found using a nodal admittance method and is made without timestep delays in the solution.

Figure 4 shows the fundamental principle of node tearing with the SSN method: the introduction of unknown-valued voltage sources at subsystem boundaries leads to a virtual decoupling of the partitions (called *SSN groups*), each described by its own set of internal state-space equations.

The SSN code has notably been optimized for real-time simulation. However, a key aspect of SSN is that there is a practical limit of 6 to 12 switches per SSN group. The upper limit on the switch-count depends on the number of state equations in each partition. The rationale of this limit stems from the pre-calculation of possible state-matrices in each partition to achieve real-time performance. For example, a partition having six switches requires the pre-calculation of, *at most*, 2^6=64 state matrices. This pre-calculation is repeated for each partition.

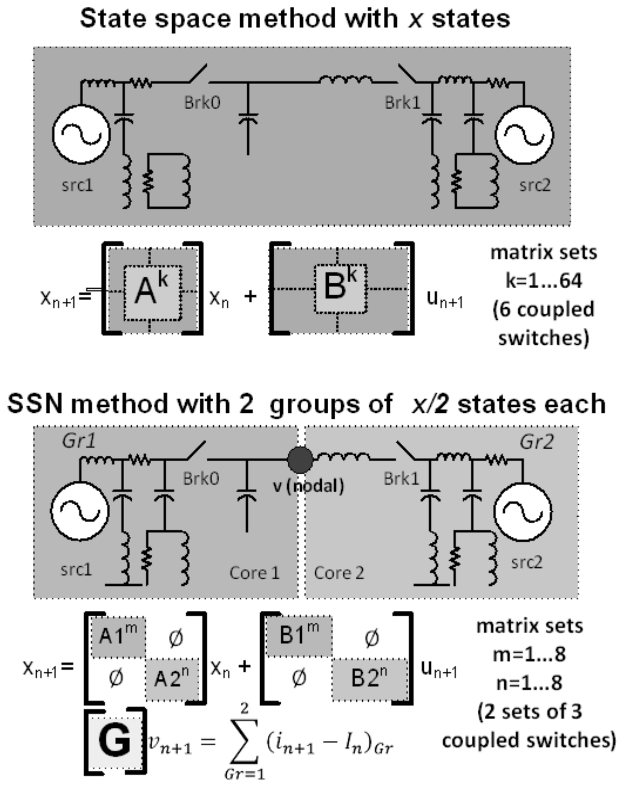
State space method with x states
Brk0
Brk1
src1
src2
$X_{n+1} = A^k X_n + B^k u_{n+1}$
matrix sets k=1...64 (6 coupled switches)
SSN method with 2 groups of $x/2$ states each
Gr1
Gr2
v (nodal)
Core 1
Core 2
$X_{n+1} = \begin{bmatrix} A1^m & \varnothing \\ \varnothing & A2^n \end{bmatrix} X_n + \begin{bmatrix} B1^m & \varnothing \\ \varnothing & B2^n \end{bmatrix} u_{n+1}$
matrix sets m=1...8 n=1...8 (2 sets of 3 coupled switches)
$G v_{n+1} = \sum_{Gr=1}^{2} (i_{n+1} - I_n)_{Gr}$

Figure 4 SSN principle showing the tearing of a node.

IV. Results

The performance and accuracy of each partition method is discussed next. Both partitioning methods are used on the same model shown in Figure 7; however, each method evaluates a different fault scenario for variety.

A. Node Tearing: Performance

The node tearing method is demonstrated by applying a three-phase fault (a-b-c, ungrounded) at the location indicated in Figure 5. This fault closes-in at 10 ms, and initiates its opening at 15 ms. The fault current, however, is not interrupted until the next zero-crossing of current in each phase.

The hardware specifications of the computer used with the node tearing algorithm is listed in TABLE II. This is a commercial off-the-shelf computer running *Windows* 7, and is used for every day computing.

TABLE II. Computer used in node tearing evaluation

Brand & Model	Dell Precision T7500
Memory (RAM)	12 GB
Operating System	Windows 7 (64-bit) with Service Pack 1
Processor	Intel Xeon E5630, 2.53 GHz, quad-core

Figure 5 conveys the performance results of the node-tearing algorithm. The blue columns show the speedup as a function of the number of partitions (p=2 to p=12). The red columns show the run time in seconds corresponding to said speedups. The green line (plotted against the right) shows the average frame time (μs) incurred by *CEMSolver*.

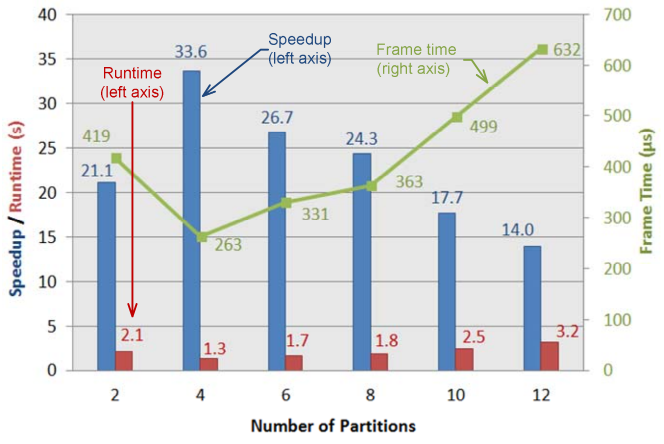

Figure 5 Performance results when using node tearing.

Speedup [28] is the ratio of time it took *MATLAB/Simulink* 2012b to complete the simulation to the time it took *CEMSolver* to complete the same simulation. The simulation was run in *MATLAB/Simulink* using both Tustin and backward Euler integration, the discrete fixed-timestep solver, and a timestep of Δt=10 μs . These times were measured from the moment each solver finished their initialization to the moment the simulation reaches its stop time of t_{stop}=25 ms. The run times (red columns) in Figure 5 represent partitioned-simulation time produced with *CEMSolver*. The runtime of *Simulink* was 44.16s seconds when using the computer described in TABLE II.

The frame time is the *average* time (μs) it took for *CEMSolver* to advance one time step. This value is computed as dh/dk, where dh is the number of microseconds the solver requires (on average) to advance dk number of steps. This relation is useful to determine the average time spent by *CEMSolver* in each timestep.

Referring to the speedup in Figure 5 , the best performance was achieved when the system was partitioned p=4 times. With this number of partitions, the simulation runtime decreased 33.6 times (from ~44 s to ~1.3 s). The maximum occurred when the number of partitions (p) matched the number of cores (c). This matching result (p=c=4) is often observed when parallelizing simulations on quad-core computers, but may not be true of computes with more than four physical cores.

The frame time in Figure 5 shows that the average time spent in each timestep in μs. This indicates that—theoretically—*CEMSolver* can be used for real time simulations having Δt=300 μs. In practice, however, this value of Δt is unacceptable as the non-determinism of *Windows* would undoubtedly vary the solution time. Furthermore, the reported frame time is an *average* time, which means that during some timesteps, the solution takes >300 μs. Nonetheless, it is interesting to note that parallel offline simulation has the potential to achieve μs timesteps in large power system model simulations.

B. Node Tearing: Accuracy

Figure 8 and Figure 9 show voltage and current at the measurement location indicated in Figure 7. The left-most column (in both figures) show the voltage (a) and current (c) produced by *MATLAB*/*Simulink* (or *SimPowerSystems*). The center column shows the voltage (b) and current (e) produced by *CEMSolver*, which uses root matching [29-31] as the integration method in both figures. The right-most columns show the absolute value of the instantaneous (point-by-point) difference in voltage and current (sub-figures (c) and current (f)).

The thick bidirectional arrows connecting sub-charts (a) and (b) or (d) and (e) (in all figures) suggest the visual impact of the result. For example, comparing Figure 8(a) and (b), there is a clear distinction between the results. However, a comparison of Figure 8(d) and (e) suggests a perceived agreement in the results. Sincevisual perceptions can be deceptive, sub-charts (c) and (f) in all figures report the actual differences. The absolute value of the instantaneous difference is calculated as $\left|x_i - x_j\right|$, where x_i represents a data point (voltage or current) produced by *Simulink* and x_j the corresponding data point produced by *CEMSolver*.

Figure 8(a) shows an initial distortion in the voltage. These oscillations are numerical chatter introduced by the Tustin (or trapezoidal rule) integration method. Figure 8(b) shows the same measurements produced in parallel with *CEMSolver*, and do not exhibit chatter. The absolute value of the instantaneous difference of these two voltages are shown in Figure 8(c). As seen, the difference in voltage reaches several hundred volts and dampens out as the simulation progresses. It should be noted that the numeric differences are artificial.

Figure 8(d) and Figure 8(e) show the current measurements at the same location. The currents appear to agree, but Figure 8(f) reveals they do not. The reasons for disagreement are the difference in integration methods, modeling differences, and the presence of numerical chatter.

The simulation results in Figure 8 were produced using the Tustin integration method in *Simulink*. A similar comparison using the backward Euler integration in *MATLAB*/*Simulink* is shown in Figure 9. The backward Euler method was used to discriminate major differences when comparing the results produced in parallel with *CEMSolver*. For example, comparing Figure 8(a) (*Simulink*, Tustin) and Figure 9 (a) (*Simulink*, backward Euler), the voltage chatter is no longer present. The absence of chatter reduces the result difference as shown in Figure 9(c). (The scale in Figure 9 (c) was adjusted with respect to Figure 8(c) for convenience). The results produced by the backward Euler technique agree more with the results produced with *CEMSolver*.

C. SSN: Performance

Acceleration with the SSN method is demonstrated by applying a three-phase fault in front of L01 (300 kW). Due to assumed relay coordination failures, neither the bus transfer at the load changed its supply to the alternate path, nor the circuit breaker at switchboard 1 (7th from the right) operated to isolate the fault. Instead, the circuit breaker at the generator (GEN1) operated to disconnect GEN1 from its switchboard. After the monetary fault cleared itself, the circuit breaker at GEN1 was re-closed.

SSN performances were obtained by running the notional power system model shown in Figure 5 with p=2 and p=6 partitions on a dual hexa-core 3.33 GHz *Linux*-based RT-LAB real-time simulator. The number of switches was limited to 6 per partition to warrant real-time simulation time frames. This was accomplished by replacing the protective devices with series resistors in their closed (1 mΩ) and open (1 MΩ) positions where appropriate.

The 885 state-variables in the notional shipboard power system model were evenly divided across the groups/partition. The partitions were executed in parallel using several cores with RT-LAB by specifying a timestep of Δt = 50 μs. The actual (measured) timestep and simulation speedups obtained with the SSN method are listed in TABLE III. It is noted that the SSN method could not achieve real-time performance for this large model. Although it was expected to execute the model at a frame time of Δt=50 μs, the measured execution frame time was, at best 280 μs, for the p=2 case. This result is similar to what was obtained in the node tearing method in Figure 5 (p=4). However, the SSN method was run in real time using a *Linux* machine while the node tearing method was run offline using a *Windows* machine.

The offline simulation produced with *SimPowerSystems* when using Tustin integration took 42 s to run the model until t_{stop}=1 s. This gives an average frame time of 2,100 μs (2.1 ms), which was used to compute the SSN real-time acceleration factor listed in TABLE III. One must also consider that the SSN timing is a strict real-time result: it means that all time steps, even at switching instants, take the same time to compute. In offline simulations with *SimPowerSystems* or the node tearing method used by *CEMSolver*, this per-step calculation time (frame time) jitters considerably. For this reason, neither *CEMSolver* nor *SimPowerSystems* can be used for real-time simulation at this time.

TABLE III. PERFORMACNE RESULTS OF THE SSN METHOD

Number of partitions	Timestep (μs)	Speedup
1	412	5.1
2	280	7.5
6	338	6.2

D. SSN: Accuracy

The accuracy of the SSN method is demonstrated by comparing the results against the results produced by *SimPowerSystems*. Figure 6 shows the simulation results of SSN superimposed on to the results produced by *SimPowerSystems*. The results appear to match well except for voltage chatter in the results produced by using Tustin integration (i.e., trapezoidal solver). This type of voltage chatter is typical of the trapezoidal solver (see [32] for example) . The *art5* solver used with the SSN method is L-stable [5], and does exhibit this problem. Although not shown, the simulation results obtained in *SimPowerSystems* with the backward Euler agree well with the SSN results.

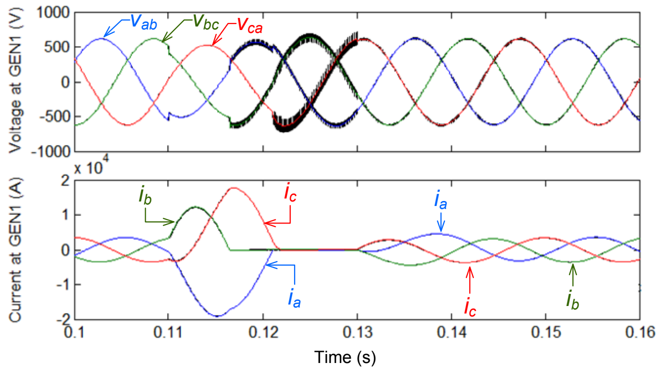

Figure 6 Accuracy comparison between *SimPowerSystems* (Tustin integration) and SSN with $\Delta t = 50\mu s$.

Conclusions

Two methods to accelerate the simulation of large power system were presented. The first method (node tearing) showed its capability to accelerate the simulation of *MATLAB*/*Simulink* model by ~30x. However, this result is not general. The speedup is highly dependent on model complexity, number of partitions, how well the partitions are defined, programming efficiency, and timestep size.

The accuracy of the node tearing method agreed more with the backward Euler results than it did with the Tustin results. The disagreement with the Tustin method was due to numerical chatter rather than physical modeling differences. Although noticeable differences were shown when comparing the results point-by-point, the instantaneous trend of the results are in general agreement. It should be observed that the word *error* is avoided, as it is not certain which result set is correct (*Simulink*'s or *CEMSolver*s'). It is likely that neither result set is correct, as both solvers only approximate actual physical behavior.

The difference in results was due—mainly—to the difference in integration methods. *CEMSolver* uses the root-matching technique, which produces stable and smoothly damped exponential responses. Integration methods based on truncated Taylor-series (such as the trapezoidal rule) produce oscillating responses during state-variable step-changes [33]. These differences were noticed throughout the simulation: before, during, and after the three-phase a-b-c fault.

The SSN method was not was not able to simulate the shipboard power system model presented in this paper in real time. However, the SSN method was able to partition and parallelize the simulations of the model. The simulations were accelerated by factors of 7x when compared to offline simulations produced with *SimPowerSystems*. Such reduction in run time is important to test various fault scenarios (fault location, type, breaker setting, etc.), which can add up to millions of run sets when statistical methods are used [34]. The accuracy of the SSN did not show visible discrepancies when compared to the results produced by *SimPowerSystems*.

Both multicore methods to partition and parallelize simulations are suitable to accelerate fault studies in shipboard power systems. The accuracy of both methods was assessed, and they appear to be within reasonable agreement to what is currently obtained with *MATLAB*/*Simulink* 2012b.

Acknowledgment

The authors acknowledge the Electric Ship Research and Development Consortium and the Office of Naval Research for their support in the development of this work.

References

[1] F. M. Uriarte, *Multicore Simulation of Power System Transients*, 1st ed. London: IET, 2013.

[2] F. M. Uriarte, R. E. Hebner, and A. L. Gattozzi, "Accelerating the simulation of shipboard power systems," in *Grand Challenges in Modeling & Simulation*, The Hague, Netherlands, June 27 - 30, 2011.

[3] F. M. Uriarte and R. Hebner, "Development of a multicore power system simulator for ship systems," in *Electric Ship Technologies Symposium*, Alexandria, VA, April 10-13, 2011, pp. 106-110.

[4] C. Dufour, J. Mahseredjian, and J. Bélanger, "A Combined State-Space Nodal Method for the Simulation of Power System Transients," *IEEE Trans. Power Delivery,* vol. 26, pp. 928-935, 2011.

[5] C. Dufour, J. Mahseredjian, J. Bélanger, and J. L. Naredo, "An advanced real-time electro-magnetic simulator for power systems with a simultaneous state-space nodal solver," in *IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, São Paulo, Brazil, 2010.

[6] F. M. Uriarte and K. L. Butler-Purry, "Multicore simulation of an AC-radial shipboard power system," in *Power Engineering Society General Meeting*, Minneapolis, MN, July 25-29, 2010, pp. 1-8.

[7] IEEE Std 45, *Recommended Practice for Electrical Installations on Shipboard*.

[8] DoD Military Specification MIL-C-24643A (1994), *Cables and Cords, Electric, Low Smoke, For Shipboard Use, General Specification for*.

[9] DoD Military Handbook MIL-HDBK-299 (SH) (1989), *Cable Comparison Handbook - Data Pertaining to Electric Shipboard Cable*.

[10] IEEE Std 1709-2010, *Recommended Practice for 1 kV to 35 kV Medium-Voltage DC Power Systems on Ships*.

[11] "Naval Ships' Technical Manual (Ch. 320) - Electrical Power Distribution Systems (rev. 2)," ed, 1998 [Online]. Available: http://www.hnsa.org/doc/nstm/ch320.pdf.

[12] A. Adediran, H. Xiao, and K. L. Butler-Purry, "The modeling and simulation of a shipboard power system in ATP," in *International Conference on Power System Transients (IPST)*, New Orleans, USA, 2003.

[13] J. G. Ciezki and R. W. Ashton, "The Resolution of Algebraic Loops in the Simulation of Finite-Inertia Power Systems," in *IEEE International Symposium on Circuits and Systems*, 1998, pp. 342-345.

[14] L. Qi and K. L. Butler-Purry, "Reformulated Model Based Modeling and Simulation of Ungrounded Stiffly Connected Power Systems," in *IEEE Power Engineering Society General Meeting*, 2003, pp. 725-730.

[15] R. M. Hamouda, M. A. Badr, and A. I. Alolah, "Effect of torsional dynamics on salient pole synchronous motor-driven compressors," *Energy Conversion, IEEE Transaction on,* vol. 11, pp. 531-538, 1996.

[16] T. J. McCoy, "Dynamic Simulation of Shipboard Electric Power Systems," Master's of Science Thesis, Dept. of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, MA, 1993.

[17] K. L. Butler-Purry, N. D. R. Sarma, C. Whitcomb, H. D. Carmo*, et al.*, "Shipboard Systems Deploy Automated Protection," *IEEE Computer Applications in Power,* vol. 11, pp. 31-36, Apr. 1998.

[18] J. S. Mayer and O. Wasynczuk, "An Efficient Method of Simulating Stiffly Connected Power Systems with Stator and Network Transients Included," *Power Systems, IEEE Trans.,* vol. 6, pp. 922-929, 1991.

[19] The MathWorks, Inc. (2010). *Simulink 7 User's Guide.* [Online]. Available: http://www.mathworks.com/help/toolbox/simulink/

[20] N. H. Doerry, "Next generation integrated power systems for the future fleet," in *Corbin A. McNeill Symposium*, Annapolis, MD, 2009.

[21] F. M. Uriarte, "Multicore simulation of an ungrounded power system," *IET Electrical Systems in Transportation,* vol. 1, pp. 31-40, Mar. 2011.

[22] S. Toub. (2010). *Patterns of Parallel Programming - Understanding and applying parallel patterns with the .NET Framework 4 and Visual C#.* [Online]. Available: http://www.microsoft.com/en-us/download/details.aspx?id=19222

[23] G. C. Hillar, *Professional parallel programming with c#*, 1st ed. Indianapolis, IN: Wiley Pub., Inc., 2010.

[24] A. Brameller, M. N. John, and M. R. Scott, *Practical Diakoptics for Electrical Networks*. London: Chapman & Hall, 1969.

[25] J. R. Marti, L. Linares, J. A. Hollman, and F. A. Moreira, "OVNI: Integrated Software/Hardware Solution for Real-Time Simulation of Large Power Systems," in *Power Systems Computation Conference (PSCC'02)*, Sevilla, Spain, 2002, pp. 1-7.

[26] G. Kron, *Diakoptics: The Piecewise Solution of Large-Scale Systems*. London: MacDonald & Co., 1963.

[27] A. Klos, "What Is Diakoptics?," *International Journal of Electrical Power & Energy Systems,* vol. 4, pp. 192-195, 1982.

[28] A. Y. Zomaya, *Parallel and Distributed Computing Handbook*. New York: McGraw-Hill, 1996.

[29] N. R. Watson and G. D. Irwin, "Electromagnetic transient simulation of power systems using root-matching techniques," *IEE Proceedings: Generation, Transmission and Distribution,* vol. 145, pp. 481-486, 1998.

[30] N. Watson and J. Arrillaga, *Power Systems Electromagnetic Transients Simulation*, 1st ed. London: IEE, 2003.

[31] J. M. Smith, *Mathematical Modeling and Digital Simulation for Engineers and Scientists*, 2nd ed. Washington, DC: John Wiley, 1987.

[32] J. R. Marti and J. Lin, "Suppression of numerical oscillations in the EMTP," *IEEE Trans. Power Systems,* vol. 4, pp. 739-747, 1989.

[33] R. M. Kielkowski, *Inside SPICE*, 2nd ed. New York: McGraw-Hill, 1998.

[34] J. Belanger, P. Venne, and J.-N. Paquin, "International Conference on Renewable Energies and Power Quality (ICREPQ'10)," in *IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, Granada (Spain), March 23-25, 2010.

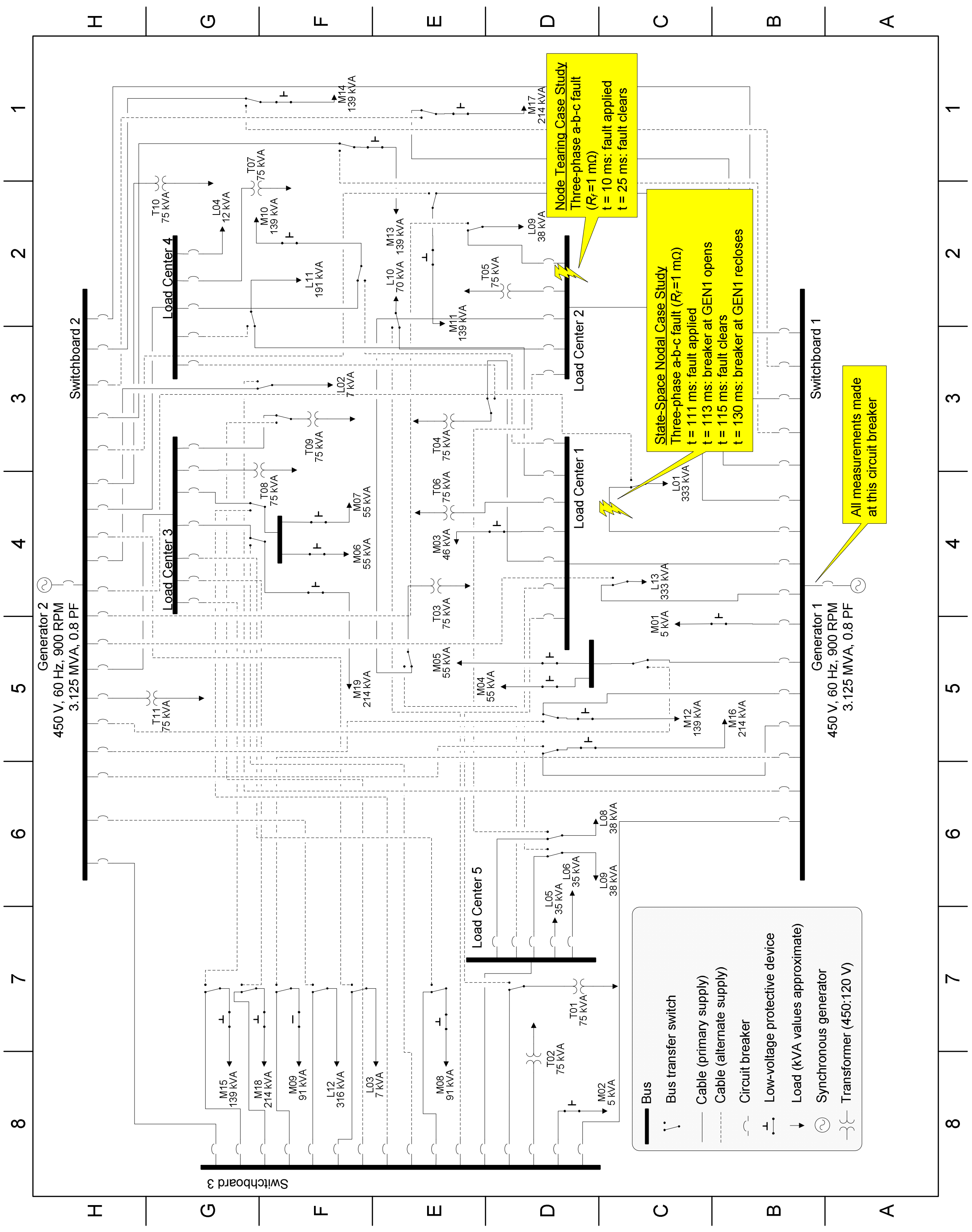

Figure 7 One-line diagram of notional shipboard power system (AC-radial, 450 V, 60 Hz, three-phase, delta-ungrounded)

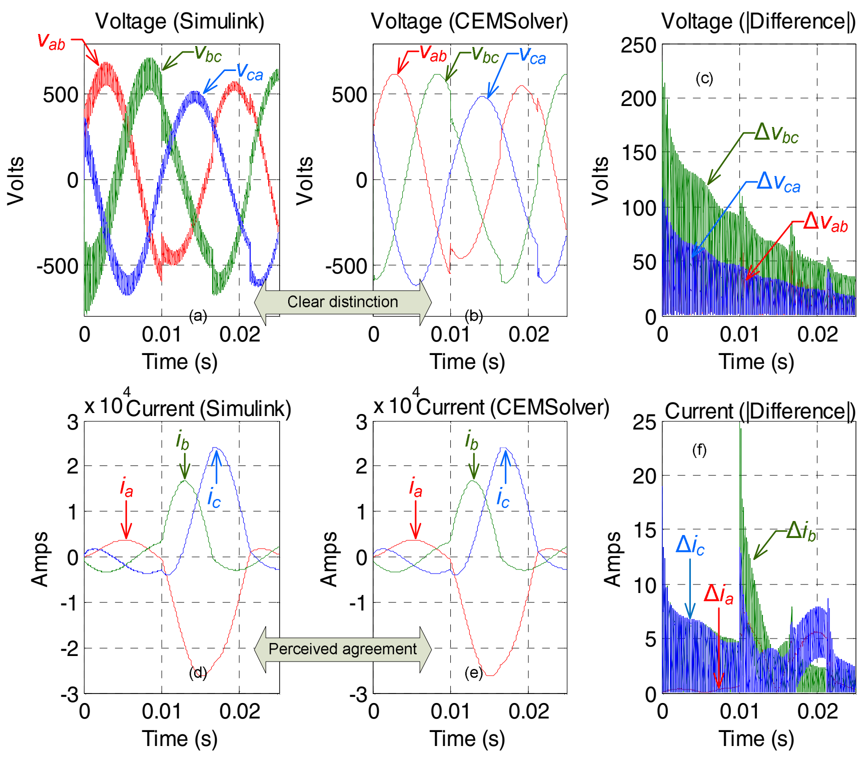

Figure 8 Node tearing: result comarison against *Simulink*'s *Tustin* integration method

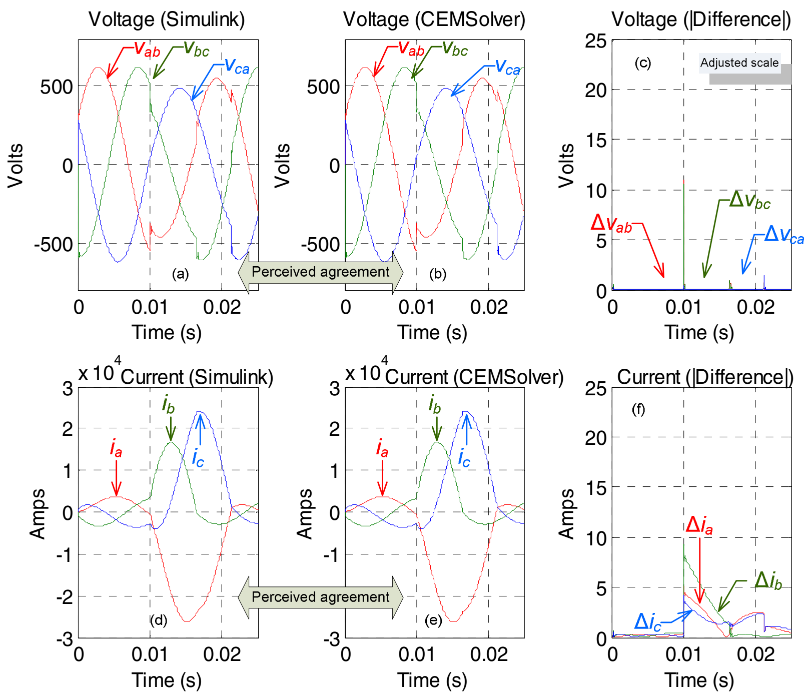

Figure 9 Node tearing: result comarison against *Simulink*'s *backward Euler* integration method