ACCELERATING THE SIMULATION OF SHIPBOARD POWER SYSTEMS

By:

F.M. Uriarte
R.E. Hebner
A.L. Gattozzi

Grand Challenges in Modeling & Simulation (GCMS2011), Part of the 2011 Summer Simulation Multiconference 2011 (Summer Sim2011), The Hague, Netherlands, June 27-30, 2011

PR 515

Center for Electromechanics
The University of Texas at Austin
PRC, Mail Code R7000
Austin, TX 78712
(512) 471-4496

Accelerating the Simulation of Shipboard Power Systems

F. M. Uriarte
f.uriarte@cem.utexas.edu

R. E. Hebner
r.hebner@cem.utexas.edu

A. L. Gattozzi
a.gattozzi@cem.utexas.edu

Center for Electromechanics
The University of Texas at Austin

Keywords: multicore, parallel, partitioning, power, ship

Abstract

This paper presents an approach to accelerate the simulation of shipboard power systems. A common issue in the ship research community is the simulation run time of large-scale shipboard power systems. A contributor to lengthy run times is that PC-based power system simulators do not exploit multicore technology, which stems from the fact that software trails advances in hardware. The Center for Electromechanics at the University of Texas at Austin (UT-CEM) is developing a multicore power system solver (*CEMS*) to simulate shipboard power system models in significantly less time. The simulation run time is reduced by first partitioning power system models and then creating simulations at the subsystem level. This accelerates the simulation of shipboard power systems. A general overview of *CEMS* and a comparison of its run times against those of *SimPowerSystems* are presented. The results demonstrate that *CEMS* can speed up the simulation of shipboard power system models built with *SimPowerSystems* up to 80 times, and that, despite common belief, multicore desktop computers *are* capable of executing complex simulation scenarios.

1. INTRODUCTION

As the all-electric ship program continues to progress from the conceptual stage toward practical implementation, the need for accurate modeling and simulation of its electrical power system has become more urgent. The large assortment of loads encountered on a warship, ranging from continuous duty loads to highly intermittent ones, makes it imperative to understand how the loads can be integrated in the shipboard power system and how they will interact dynamically with each other, as well as with the available power sources. Furthermore, the increasing presence of power electronic stages, especially at the interface between the ac and dc zones of the system, requires more stringent control methods, both at the local and global level, which adds to the complexity of the system.

Under the sponsorship of the Office of Naval Research, researchers at the University of Texas Center for Electromechanics have attempted to model a notional electric ship with a collection of different loads [1]. Using *SimPowerSystems* [2], this effort resulted in the development of a relatively large power system model for a ship system supporting several load types. Although the effort produced usable and interesting results, the large power system model highlighted a major challenge: even under the assumption of a simplified architecture, the power system complexity posed strong computational demands on desktop computers, which led to unbearably long simulation run times. In fact, it was not unusual to wait one week to obtain calculations covering only six seconds of simulated data [1].

One option to speed up the simulation of power systems, and deemed to be promising in the near future, is the use of external processors designed around field programmable gate arrays (FPGA) to assist the main computer in speeding up the most intensive time-consuming calculations. Researchers at UT-CEM believes that this approach promises the largest gains in terms of speeding up computation times [3],[4], ultimately making real time simulation possible at an affordable cost, but also recognizes that substantial development effort must be allocated to capitalize on this opportunity and bring it into a practical and easy to use form. Therefore, while work on this front continues, it is advisable to explore ways to optimize the use of existing hardware resources by improving our software tools. In fact software—not hardware—is the major restriction on computing performance at the present time. A clear testimony of this restriction can be gathered from observing how the hardware commonly available in present day PCs is used inefficiently by existing commercial codes.

Other efforts to speed up the simulation time of shipboard power systems exist as well, but are not widely available or come at a cost. In [5], the latency insertion method was used to partition a large, notional electric shipboard power system into subsystems. Although speed gains of 14x were reported on a multicore computer, the speed gain comparisons were made by comparing VTB [6] against itself instead of widely available simulators *PSCAD* or *SimPowerSystems*. Power system models in the ship community already exist in *SimPowerSystems* or PSCAD [7] and remaining in such environments is highly desirable.

A well known real time solver is Opal-RT's ARTEMIS [8], which is currently capable of importing large shipboard power systems models from *SimPowerSystems* and solve them in real-time when using their target hardware. A solution as such, however, requires the acquisition of costly specialized hardware and is not accessible to all members of the ship research community.

This paper presents an overview of UT-CEM's efforts to develop fast power system simulations for Navy entities. Section 2 examines the problem of slow simulation. Section 3 describes the solution methodology currently implemented by *CEMS*. Section 4 analyzes three case studies to assess the speed-ups of *CEMS* when comparing to *SimPowerSystems*. Section 5 discusses the results, Section 6 a summary of the work, and Section 7 concludes.

2. PROBLEM STATEMENT

The problem of lengthy simulation can be explained by referring to the loop listed as (1). This loop is known as the *time loop* and is common to several commercial simulators [9]. The variable t represents the simulation time, and parameters Δt and t_{end}, the time step increment and simulation stop (end) time, respectively. Typically the time parameters are constrained to $\Delta t < 50\ \mu\text{s}$, $t_{end} < 5\ \text{s}$, but depend on the transients of interest, time constants of the power system, and the time-spacing between staged events [1].

$$\begin{cases} \text{for } t = 0;\ t < t_{end};\ t = t + \Delta t \\ \qquad \text{re-factor } \mathbf{A}(t) & (a) \\ \qquad \text{solve } \mathbf{A}(t)\mathbf{x}(t) = \mathbf{b}(t) & (b) \\ \qquad \text{interpolate } \mathbf{x}(t) & (c) \\ \text{end} \end{cases} \tag{1}$$

The time loop in (1) also highlights the most time consuming subroutines: a) refactoring the network coefficient matrix **A** when the coefficients change, b) the solution of a sparse system of equations of the form **A**·**x** = **b**, and c) the interpolation of the solution vector when switching events between t and $t + \Delta t$ are missed. (Other subroutines are elided from (1) as they are relatively fast when compared to the ones shown.)

While (1)(c) does not require much time itself, it does not allow the simulation time to advance from t to $t + \Delta t$ until *all* events occurring between t and $t + \Delta t$ are captured. This situation may require solving the system several times at t before the simulation time can advance. (Time interpolation [10],[11] occurs frequently in the presence of multiple three-phase rectifiers (e.g., > 15) and constitutes a major, yet implicit, overhead.)

Referring to (1), it is not difficult to show that the simulation of a power system can take 24 hours to complete. Consider, as an example, the case when $\Delta t = 5\ \mu\text{s}$ and $t_{end} = 5\ \text{s}$. These parameters require executing the time loop in (1) $5 / 5\times 10^{-6} = 10^{6}$ times. If, say, the average time loop iteration takes 86.4 ms, which is apparently fast, the total simulation run time is $10^{6} \times 86.4 \times 10^{-3} = 24$ hours, and is impractical in terms of man- and machine-hours.

This paper will show that *CEMS* spends considerably less average time in (1) than *SimPowerSystems* does.

3. SOLUTION METHODOLOGY

CEMS reduces the computational burden observed in (1) in two stages: by *partitioning* the power system model and then by *reformulating* the network equations. Before addressing these stages the user interface used by *CEMS* is briefly described.

3.1. User Interface

CEMS uses the *Simulink* user interface [2],[12]. This (central) trait makes *CEMS* capable of importing existing *SimPowerSystems* models [13]. This capability further keeps users from redrawing large models which takes considerable time. In addition, this time-saving feature does not require users to learn an unfamiliar interface, and allows them to continue using and sharing existing *SimPowerSystems* models with others in the ship research community.

3.2. Partitioning

The motive behind partitioning power systems is creating subsystems that require less computational effort to iterate through (1) than the original unpartitioned power system. Partitioning power systems, however, requires knowledge of *where* to tear and of *how many* partitions to create. The answers to these questions can be difficult to get right.

As to where to tear, the problem of locating the best disconnection points of a power system is an NP-complete problem and cannot be expressed in closed form [14-16]. This task is currently outsourced to graph-partitioning software *hMetis* [17]. By invoking *hMetis* with a representative graph of the power system, *hMetis* returns p vertex sets where p represents the number of specified partitions.

As to how many partitions to create, p is currently matched to the number of cores, which in this paper is four. Although matching the number of partitions to the number of cores is "simplistic in its analysis" [18], it has returned acceptable speed gains so far.

To illustrate how *CEMS* invokes *hMetis* to partition a power system, consider the power system shown in Figure 1. This power system is a simple, scaled-down system, used for illustration purposes only. In Figure 1, the power apparatus are represented as multi-terminal components [19], interconnected at three-phase nodes. *CEMS* maps this power system to a graph, where each vertex represents a power apparatus and each edge represents a *junction*. (The number of phase conductors at each junction is currently irrelevant). The graph corresponding to the power system of Figure 1 is shown in Figure 2.

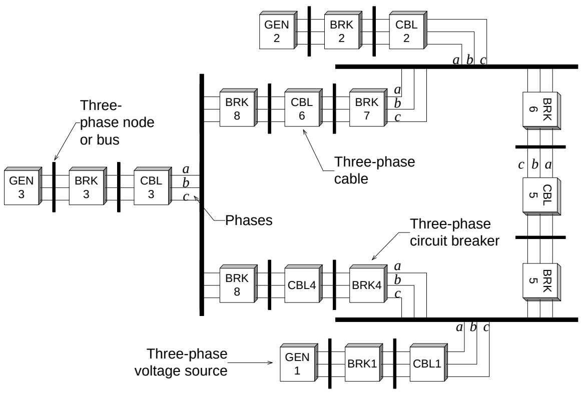

Figure 1. Original (unpartitioned) power system

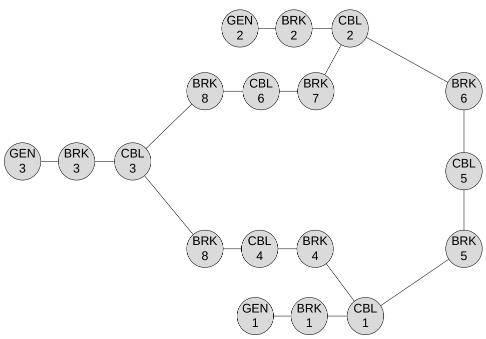

Figure 2. Graph representation of the power system

The representative graph is passed as a text file to *hMetis* [17], which has readily-implemented partitioning and balancing heuristics [20] suitable for locating the disconnection points of graphs [21]. The output produced by *hMetis* is shown graphically in Figure 3 as two sets of vertices where each vertex *set* is one graph partition.

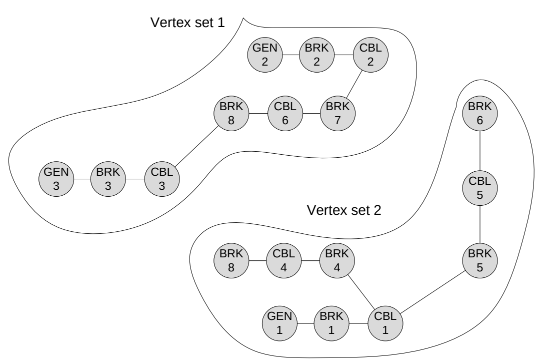

Figure 3. Graph as partitioned by *hMetis.*

The results of *hMetis* are passed back to *CEMS* as a text file. *CEMS* interprets the results produced by *hMetis* (i.e., the vertex sets) as shown in Figure 4, which reveals the locations of the disconnection points.

When the disconnection points are known, *CEMS* assembles the system matrices according to a reformulated system of equations which is different than **A**·**x** = **b** shown in (1). The reformulated matrices are described in the next subsection.

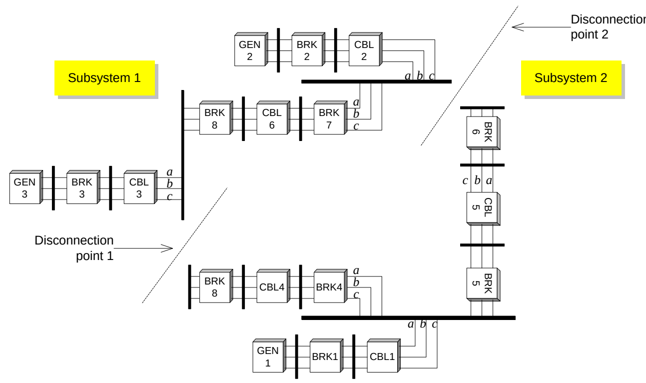

Figure 4. Partitioned power system according to its representative graph: two subsystems created from two disconnection points

3.3. Network Reformulation

Consider the electrical network at a disconnection point shown in Figure 5. At this disconnection point, *CEMS* inserts in-line current sources of *unknown* value in each phase. From current source transportation [22], [23], these current sources can be torn as two current sources per phase: one current source that sinks subsystem 1 into the ground, and the other that acts as a source to subsystem 2. The boundary injections i_a, i_b, and i_c are the same in both subsystems, but are unknown. Referring to Figure 5, when the subsystems are formed, the intersection voltages v_a, v_b, and v_c are bisected as $\{v_{a1}, v_{a2}\}$, $\{v_{b1}, v_{b2}\}$, and $\{v_{c1}, v_{c2}\}$, respectively. This partitioning technique is known as *node tearing* [24-30].

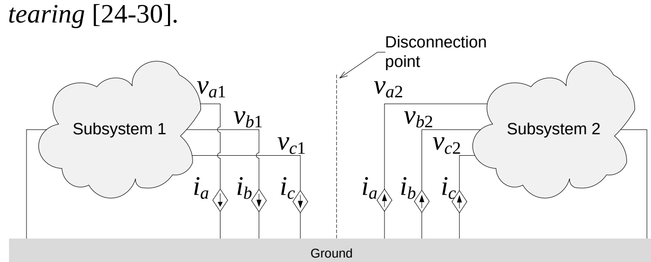

Figure 5. Electrical network (node) tearing once the disconnection points are known

Since the subsystems are physically isolated, they can be arranged in block-diagonal form in a new compound network matrix. The coupling between subsystems is now through the current source variables arranged in a vector **u**. To link the variables of each subsystem to **u**, matrices containing only ± 1 s and 0 s are used. These matrices, noted $\mathbf{D}_i$ for the i^{th} subsystem, couple (i.e., link [31],[32]) the subsystem variables to the boundary variables so that numerical coupling between subsystems is maintained.

The new reformulated system of equations is given in (2) for p subsystems (or partitions). The number of boundary variables in $\mathbf{u}$ varies with the system topology, number of partitions, and formulation method (i.e., nodal analysis in this case).

$$\left[\begin{array}{cccc|c} \mathbf{A}_1 & \cdot & \cdot & \cdot & \mathbf{D}_1 \\ \cdot & \mathbf{A}_2 & \cdot & \cdot & \mathbf{D}_2 \\ \cdot & \cdot & \ddots & \cdot & \vdots \\ \cdot & \cdot & \cdot & \mathbf{A}_p & \mathbf{D}_p \\ \hline \mathbf{D}_1^{\mathrm{T}} & \mathbf{D}_2^{\mathrm{T}} & \cdots & \mathbf{D}_p^{\mathrm{T}} & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_p \\ \hline \mathbf{u} \end{array}\right] = \left[\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_p \\ \hline \mathbf{0} \end{array}\right] \tag{2}$$

$\mathbf{A}_i$	= nodal conductance matrix for subsystem i
$\mathbf{b}_i$	= current injection vector for subsystem i
$\mathbf{D}_p(i,j)$	= connection matrix linking the p^{th} subsystem to its boundary variables. The matrix elements are: $\begin{cases} =1, & \text{if current source } j \text{ sinks from node } i \\ =-1, & \text{if current source } j \text{ injects at node } i \\ =0, & \text{elsewhere} \end{cases}$
p	= number of partitions
$\mathbf{u}$	= vector of boundary variables (i.e., *all* unknown boundary current sources)
$\mathbf{x}_i$	= vector of node voltages in subsystem i

The formulation in (2) can be solved as (3), which takes considerable less time than the solution of $\mathbf{A}\cdot\mathbf{x}=\mathbf{b}$ in (1) for at least three reasons [33]:

1. The computation of the first term on the right-hand side of (3) (top) can be parallelized
2. The second term on the right-hand side of (3) (top) can also be parallelized (except for the computation $\mathbf{u}$)
3. Matrices $\mathbf{A}_1$, $\mathbf{A}_2$, … , $\mathbf{A}_p$ are much smaller than $\mathbf{A}$. Referring back to (1), working concurrently with smaller matrices $\mathbf{A}_i$ reduces significantly the average time spent during in the time loop.

$$\begin{cases} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1^{-1}\mathbf{b}_1 \\ \mathbf{A}_2^{-1}\mathbf{b}_2 \\ \vdots \\ \mathbf{A}_p^{-1}\mathbf{b}_p \end{bmatrix} - \begin{bmatrix} \mathbf{A}_1^{-1}\mathbf{D}_1 \\ \mathbf{A}_2^{-1}\mathbf{D}_2 \\ \vdots \\ \mathbf{A}_p^{-1}\mathbf{D}_p \end{bmatrix}\mathbf{u} \\ \mathbf{u} = \left(\mathbf{D}^{\mathrm{T}}\mathbf{A}_{block}^{-1}\mathbf{D}\right)^{-1}\left(\mathbf{D}^{\mathrm{T}}\mathbf{A}_{block}^{-1}\mathbf{b}\right) \end{cases} \tag{3}$$

It should be pointed out that although $\mathbf{u}$ in (3) is the serial part of the parallel solution, it does not constitute pronounced overhead when the simulations are coarse-grained [34]. In matrix theory (3), it is known as Woodbury's method for inverting modified matrices [30],[35],[36], which is also of the form of Kron's diakoptics [36],[37], and Ho's modified nodal analysis[38],[39]. The derivations of (2) and (3) can be found in [21],[30],[40].

4. CASE STUDIES

Here performance results are presented by comparing the run times of *CEMS* and *SimPowerSystems*. While it is understood that the speed gain depends on the power system being simulated [5], it is not possible to simulate and compare the run times of all power systems. To allow readers to assess the physical and numerical complexity of the case studies presented, each case study is accompanied by power apparatus and topology enumerations.

The power system examined here is based on a notional large-scale ac-radial shipboard power system [41], which was arbitrarily chosen to compare the run times of *SimPowerSystems* and *CEMS*. This power system represents a 450 VAC ac-radial shipboard power system with ring topology and is considered to have necessary (though not sufficient) complexity to assess simulation speed-ups.

In all case studies, *SimPowerSystems* and *CEMS* were run on the computer listed in Table 1. This computer is currently available to researchers, implying that the speed gains observed here are likely to be similar on other desktop computers. In all cases the time parameters were set to $t_{end}=0.1\text{ s}$ and $\Delta t = 50\ \mu\text{s}$. When executing the simulations with *CEMS*, the number of partitions was held constant at p=4.

Table 1. Computer implementation details

Brand & model	Dell Precision T7500
Processor	Intel Xeon E5630, 2.53 GHz, quad-core
Physical memory	12 GB
Operating system	*Windows* 7 Professional with service pack 1 (64-bit)
Programming language	*Microsoft C#* 4.0 with .NET 4.0

4.1. Speed-up

The speed-up (or speed gain) is computed with (4), where the numerator represents the run time (in seconds) of *SimPowerSystems*, and the denominator, the partitioned run time of *CEMS* (in seconds). It is important to note that the authors have no control over the numerator in (4). This makes (4) a fair comparison as both programs will independently attempt to reduce their contributions to (4). Furthermore, it should be noted that (4) only compares *run time* and not initialization time. Both *SimPowerSystems* and *CEMS* spend about a minute initializing which is not accounted for in (4). In the case of *CEMS*, initialization includes calling *hMetis*, which takes < 1 s as the representative graphs created by *CEMS* are small (e.g., < 400 vertices, and < 400 edges).

$$\text{Speedup} = \frac{\textit{SimPowerSystems}\ \text{run-time}}{\textit{CEMS}\ \text{run-time with } p = 4} \quad (4)$$

4.2. Accuracy

An initial assessment of *CEMS*' accuracy was presented in [21]. This paper focuses on an initial assessment of *speed* for which accuracy is considered outside the scope of this writing. Understanding, however, that accuracy is more important than speed, the authors are currently developing a program that will compare the (hundreds of thousands of) data points generated with *SimPowerSystems* and *CEMS*.

The average and maximum errors will be computed with (5) where $avg_p(\ )$ is a function that calculates the average error between *SimPowerSystems* (unpartitioned) and all *CEMS* partitioned simulations. In (5), x_{unp}^{k+1} represents a data point generated by *SimPowerSystems* (e.g., an instantaneous voltage), and x_{par}^{k+1} is the corresponding data point generated by *CEMS*. Likewise, $max_p(\ )$ will be used to calculate the maximum error between unpartitioned and partitioned simulations.

$$\text{avg}_p(\ ) \text{ or } \max_p(\ ) \rightarrow \left(100 \left| \frac{x_{unp}^{k+1} - x_{par}^{k+1}}{x_{unp}^{k+1}} \right| \% \quad \forall \quad x_{unp}^{k+1} \right) \quad (5)$$

4.3. Description of Case Studies

Three variations of the *same* power system are presented as three cases studies. Each variation is described in terms of the component count and topology information as output by the *SimPowerSystems* netlist feature.

4.3.1. Case 1

This is the most comprehensive (and important) simulation case. This case includes all the power apparatus and power converters of the shipboard power system. To assess the *physical* complexity of this system, the *SimPowerSystems* block-counts are enumerated in Table 2. To assess the *numerical* complexity of this system, the information provided by *SimPowerSystems* netlist feature is provided in Table 3. Both tables can be used to compare the system presented here with other systems currently being benchmarked [42] in the ship research community. (Similar tables are provided for cases 2 and 3.)

4.3.2. Case 2

This case removes all power apparatus (except power converters) from Case 1. This change reduced the RLC branch count while maintaining the switch count constant. The power apparatus count and netlist output for this case are given in Table 4 and Table 5, respectively.

4.3.3. Case 3

This case replaces the 19 three-phase rectifiers, DC buses, and three-phase inverters of Case 1 with static three-phase RL loads. All loads, formerly driven by inverters, are directly connected to the 450 VAC mains in this case. (i.e., there are no power electronic switches; see Table 6 and Table 7.) Having three variations of the *same* system will permit determining whether the speed gain varies with the RLC and switch counts.

Table 2. *SimPowerSystems* block count for Case 1

Series RLC branch	19
Series RLC load	11
Three-phase circuit breaker	158
Three-phase measurement	5
Three-phase PI section line	108
Three-phase series RLC load	43
Three-phase source	3
Three-phase transformer (two-winding)	11
Universal bridge as a three-phase inverter with PWM controller)	19
Universal bridge as an uncontrolled three-phase rectifier	19
Total	396

Table 3. *SimPowerSystems* netlist output for Case 1

Electrical nodes	1,209
Branches	2,254
State-variables	1,549
Switches*	702

**SimPowerSystems* includes protective devices in the switch count

Table 4. *SimPowerSystems* block count for Case 2

Series RLC branch	38
Series RLC load	0
Three-phase circuit breaker	60
Three-phase measurement	4
Three-phase PI section line	40
Three-phase series RLC load	19
Three-phase source	3
Three-phase transformer (two-winding)	0
Universal bridge as a three-phase inverter with PWM controller	19
Universal bridge as an uncontrolled three-phase rectifier	19
Total	202

Table 5. *SimPowerSystems* netlist output for Case 2

Electrical nodes	544
Branches	992
State-variables	658
Switches	408

5. RESULTS

The run times and speed-ups for the all three cases are shown in Table 8 and Figure 6. The average time (per iteration) spent on the time loop in (1) of each simulator is compared in Table 9. The results for the first case are the most significant results, as they represent a simulation of the *whole* system. As both *SimPowerSystems* and *CEMS* ran on the same computer, a speed-up of 77x confirms that the bottleneck of simulation is currently in the software and not the hardware.

Table 6. *SimPowerSystems* blocks for Case 3

Series RLC branch	0
Series RLC load	0
Three-phase circuit breaker	159
Three-phase measurement	3
Three-phase PI section line	107
Three-phase series RLC load	43
Three-phase source	3
Three-phase transformer (two-winding)	11
Universal bridge as a three-phase inverter with PWM controller	0
Universal bridge as an uncontrolled three-phase rectifier	0
Total	326

Table 7. *SimPowerSystems* netlist output for Case 3

Electrical nodes	1,108
Branches	1,995
State-variables	1,405
Switches	474

Case 2 presented a reduced version of Case 1 where both the electrical node and RLC branch counts were reduced. From Table 3 and Table 5, Case 2 presented $544/1209 \approx 44\%$ of the original electrical node count and $658/1549 \approx 42\%$ of the state-variable count. This network reduction seems to have favored *SimPowerSystems* in that the speed-up reduced to 26x. This result may be suggestive that small power systems may be better suited for *SimPowerSystems*. Nevertheless, the speed-ups obtained with *CEMS* for this system exceed an order of magnitude.

In Case 3, which is a pure RLC (passive) system with no power electronic switches, the gains dropped from 77x to 47x. Although the gain observed is acceptable, the reduction in speed-up is noticeable.

6. SUMMARY

This paper presented in-progress efforts and results of *CEMS*. The problem of time domain simulations was determined to be the time loop, where operating on large matrices (i.e., refactoring and solving) takes considerable time when the number of time step approaches $O(10^6)$.

CEMS does not have a user interface. Instead, *CEMS* imports (select) power system models built using blocks from the *SimPowerSystems* blockset in *Simulink*. This trait makes *CEMS* flexible in that users can remain in a familiar working environment and continue to grow existing models without being hindered by lengthy execution times as currently experienced with the *MATLAB*/*Simulink* solver.

Partitioning in *CEMS* requires using graph theory and node tearing. Graph theory is invoked to determine the location of the disconnection points. This is a difficult task currently outsourced to *hMetis*. Once the best possible disconnection points are found, *CEMS* forms subsystem-level matrices. Node tearing is more flexible than diakoptics [37],[40] in that boundary branches do not have to exist at the boundaries between subsystems in order to partition a network [43]. This reduces the size of the graph and relaxes the partitioning constraints imposed on *hMetis*.

Table 8. Run time comparisons (seconds)

	SimPowerSystems 5.2	*CEMS* (beta)	Speedup
Case 1	1,376	18	77
Case 2	86	3	26
Case 3	28	1	47

Table 9. Average time (seconds) spent in (1)

	SimPowerSystems 5.2	*CEMS* (beta)
Case 1	688.0E-3	9.0E-3
Case 2	43.0E-3	1.7E-3
Case 3	14.2E-3	0.3E-3

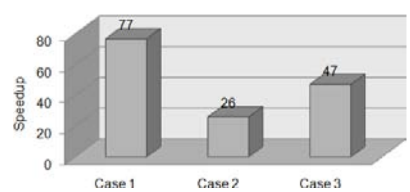

Figure 6. Speed-ups when comparing the run times of *SimPowerSystems* and *CEMS* (*p*=4).

Three case studies were presented. Each case study had a different number of state-variables, power apparatus, and electrical nodes. In the most comprehensive case (Case 1), the speed-ups were the most significant. Cases 2 and 3 had less complexity than Case 1. The gains in these latter cases did not match the speed-ups observed in Case 1; however, in all cases the speedups are acceptable.

The reason for uneven speed-ups has not been thoroughly studied by the authors, but may be associated with Amdahl's law [44]. It is possible that for smaller electrical-node count power systems, the degree of parallelism in *CEMS* is comparable to its degree of serialism; this limits—but does not hinder—the speed-ups. In the case of large power systems, and from the results presented, the degree of parallelism clearly outweighed the serialism and resulted in significant speed-ups. In this regard, it is possible that *SimPowerSystems* may be better suited to simulate small (and less complex) power systems than is *CEMS*. Stated differently, *CEMS* may be better suited for larger (and more complex) power systems than *SimPowerSystems*.

7. CONCLUSIONS

The statements made here are case-specific, not general. It is possible that other shipboard power system models exhibit less speed-up and others more. At this stage of development, it is not certain whether *CEMS* will reduce the simulation run time of *every* power system model—although it appears it may. Only thorough testing of different power system models can answer such question.

Based on the authors' experience, small power systems should be simulated in *SimPowerSystems* using the *MATLAB*/*Simulink* solver. The *MATLAB*/*Simulink* solver is swift for small power system, which does not warrant using an external solver such as *CEMS*.

In the development of future electric-ship design, it is highly desirable to simulate power system models that are as large as possible and with the fewest simplifying assumptions. These types of shipboard power system models are large, complex, and time-consuming to simulate. This situation, however, *does* justify using an external solver such as *CEMS*.

The possible speed-ups with *CEMS* are clear. While the speed-ups appear to be inconsistent among themselves, they exceed one order of magnitude in all cases. These speed-ups are deemed highly desirable [1],[45] and constitute an advance toward accelerating the shipboard power system simulations.

One of the most important traits of *CEMS* is its ability to import select *SimPowerSystems* models. This particular trait will allow future users to retain (and grow) models created in the *SimPowerSystems* and optionally use *CEMS* when (and if) run times become unbearable.

REFERENCES

[1] R. Hebner, J. Herbst, and A. Gattozzi, "Large scale simulations of a ship power system with energy storage and multiple directed energy loads," in *Proc. 2010 Grand Challenges in Modeling & Simulation (GCMS2010)*, pp. 430-435.

[2] The MathWorks, Inc. (2010). *SimPowerSystems 5 User's Guide.* [Online]. Available: http://www.mathworks.com/help/toolbox/physmod/powersys/

[3] D. Chiou, D. Sunwoo, H. Angepat, J. Kim, N. A. Patil, W. Reinhart, and D. E. Johnson, "Parallelizing computer system simulators," in *Proc. 2008 IEEE Intl. Parallel & Distributed Processing Symp.*, pp. 1-5.

[4] R. Crosbie, J. Zenor, R. Bednar, D. Word, and N. Hingorani, "Using attached processors to achieve high-speed real-time simulation," in *Proc. 2010 2nd Intl. Conf. on Advances in Syst. Simul*, pp. 140-143.

[5] Y. Zhang, R. Dougal, B. Langland, J. Shi, and J. Tang, "Method for partitioning large system models when using latency insertion method to speed network solution," in *Proc. 2009 Grand Challenges in Modeling and Simulation*.

[6] E. Broughton, B. Langland, E. Solodovnick, and G. Croft. (2003). *Virtual Test Bed User's Manual.* Columbia, SC: University of South Carolina.

[7] Manitoba HVDC Research Center. (2010, Jan.). [Online]. Available: https://pscad.com/products/pscad/

[8] Opal-RT Technologies Inc. (2009). *eMEGAsim and eDRIVEsim Product Information & Simulation Application Examples.* [Online]. Available: http://www.opal-rt.com/sites/default/files/technical_papers/

[9] Manitoba HVDC Research Centre, Inc. (2005). *EMTDC User's Guide - A Comprehensive Resource for EMTDC.* Manitoba, Canada: Manitoba HVDC Research Centre.

[10] A. E. A. Araujo, H. W. Dommel, and J. R. Marti, "Converter Simulations with the EMTP: Simultaneous Solution and Backtracking Technique," in *Proc. 1993 Joint International Power Conference (IEEE - NTUA) - Athens Power Tech*, pp. 941-945.

[11] P. Kuffel, K. Kent, and G. Irwin, "The Implementation and Effectiveness of Linear Interpolation Within Digital Simulation," in *Proc. 1995 International Conference on Power System Transients*, pp. 499-504.

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

[13] F. M. Uriarte and R. Hebner, "Development of a multicore power system simulator for ship systems," in *Proc. 2011 Electric Ship Technology Symposium*.

[14] H. Ding, A. A. Elkeib, and R. Smith, "Optimal Clustering of Power Networks Using Genetic Algorithms," *Electric Power Systems Research*, vol. 30, pp. 209-214, Sep 1994.

[15] C. S. Chang, L. R. Lu, and F. S. Wen, "Power system network partitioning using tabu search," *Electric Power Systems Research*, vol. 49, pp. 55-61, Feb. 15 1999.

[16] P. Zhang, J.R.Marti, and H. W. Dommel, "Network Partitioning for Real-Time Power System Simulation," in *Proc. 2005 International Conference on Power System Transients*, pp. 1-6.

[17] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: applications in VLSI domain," in *Proc. 1997 Design and Automation Conference*, pp. 526-529.

[18] B. Wagner. (2009). *More Effective C# : 50 Specific Ways to Improve Your C#.* Upper Saddle River, NJ: Addison-Wesley.

[19] H. R. Martens and D. R. Allen. (1969). *Introduction to System Theory.* New York: Charles E. Merrill Publishing Company.

[20] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell Laboratories Record*, vol. 49, pp. 291-307, 1970.

[21] F. M. Uriarte, "Multicore Simulation of an Ungrounded Power System," *IET Electr. Syst. Transp.*, vol. 1, 2011.

[22] T. L. Pillage, R. A. Rohrer, and C. Visweswariah. (1995). *Electronic Circuit & System Simulation Methods.* NY: McGraw-Hill, Inc.

[23] R. A. Rohrer, "Circuit partitioning simplified," *IEEE Trans. Circuits and Systems*, vol. 35, pp. 2-5, 1988.

[24] P. W. Aitchison and A. Klos, "Network tearing - an alternative method," *International Journal of Electrical Power & Energy Systems*, vol. 13, pp. 308-320, Dec 1991.

[25] A. Sangiovanni-Vincentelli, C. Li-Kuan, and L. O. Chua, "A new tearing approach - node tearing nodal analysis," in *Proc. 1975 IEEE International Symposium on Circuits and Systems*, pp. 143-147.

[26] C. Yue, X. Zhou, and R. Li, "Node-splitting approach used for network partition and parallel processing in electromagnetic transient smiulation," in *Proc. 2004 International Conference on Power System Technology*.

[27] B. M. Zhang, N. A. Xiang, and S. Y. Wang, "Unified piecewise solution of power-system networks combining both branch cutting and node tearing," *International Journal of Electrical Power & Energy Systems*, vol. 11, pp. 283-288, Oct 1989.

[28] A. Kalantari and S. M. Kouhsari, "An Exact Piecewise Method for Fault Studies in Interconnected Networks," *International Journal of Electrical Power & Energy Systems,* vol. 30, pp. 216-225, 2008.
[29] S. Esmaeili and S. M. Kouhsari, "A Distributed Simulation Based Approach for Detailed and Decentralized Power System Transient Stability Analysis," *Electric Power Systems Research,* vol. 77, pp. 673-684, 2007.
[30] I. S. Duff, A. M. Erisman, and J. K. Reid. (1986). *Direct Methods for Sparse Matrices.* Oxford: Oxford University Press.
[31] 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 *Proc. 2002 Power Systems Computation Conference (PSCC'02),* pp. 1-7.
[32] J. R. Marti, L. R. Linares, J. Calviño, H. W. Dommel, and J. Lin, "OVNI: An Object Approach to Real-Time Power System Simulators," in *Proc. 1998 International Conference on Power System Technology (Powercon'98).*
[33] Z. Quming, S. Kai, K. Mohanram, and D. C. Sorensen, "Large Power Grid Analysis Using Domain Decomposition," in *Proc. 2006 Design, Automation and Test in Europe, 2006. DATE '06. Proceedings,* pp. 1-6.
[34] C. H. Lai, "Diakoptics, Domain Decoupling and Parallel Computing," *The Computer Journal,* vol. 37, pp. 840-846, 1994.
[35] H. V. Henderson and S. R. Searle, "On Deriving the Inverse of a Sum of Matrices," *SIAM Review,* vol. 23, pp. 53-60, 1981.
[36] A. Klos, "What Is Diakoptics?," *International Journal of Electrical Power & Energy Systems,* vol. 4, pp. 192-195, 1982.
[37] G. Kron. (1963). *Diakoptics: The Piecewise Solution of Large-Scale Systems.* London: MacDonald & Co.
[38] C.-W. Ho, "The modified nodal approach to network analysis," in *Proc. 1974 IEEE International Symposium on Circuits and Systems,* pp. 505-509.
[39] C.-W. Ho, A. E. Ruehli, and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. Circuits and Systems,* vol. CAS-22, pp. 504-509, Jun. 1975.
[40] A. Brameller, M. N. John, and M. R. Scott. (1969). *Practical Diakoptics for Electrical Networks.* London: Chapman & Hall.
[41] F. M. Uriarte and K. L. Butler-Purry, "Multicore simulation of an AC-radial shipboard power system," in *Proc. 2010 Power Engineering Society General Meeting.*
[42] A. Benigni, P. Bientinesi, and A. Monti, "Benchmarking different direct solution methods for large power system simulation," in *Proc. 2010 Grand Challenges in Modeling and Simulation.*
[43] M. Armstrong, J. R. Marti, L. R. Linares, and P. Kundur, "Multilevel MATE for efficient simultaneous solution of control systems and nonlinearities in the OVNI simulator," *IEEE Trans. Power Systems,* vol. 21, pp. 1250-1259, Aug. 2006.
[44] G. M. Amdahl, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities," in *Proc. 1967 AFIPS,* pp. 483-485.
[45] R. E. Hebner, A. L. Gattozzi, K. R. Cohn, and W. B. Colson, "Analysis of the power quality impact of multiple directed energy loads on an electric ship power system," in *Proc. 2010 23rd Annual Solid State and Diode Laser Technology Review.*

BIOGRAPHIES

Dr. Fabian M. Uriarte joined the Center for Electromechanics in 2010. He obtained his BS and MS in electrical engineering from Virginia Tech, and his PhD in electrical engineering from Texas A&M University in the area of parallel simulation of power systems. His research interests are in parallel computing, object-oriented programming, and modeling and simulation of power systems, power electronic converters, and microgrids.

Dr. Robert E. Hebner is the Director of the Center for Electromechanics at the University of Texas at Austin. The Center develops advanced energy technology and teams with companies to get the technology to market. Previously, he was acting Director of NIST with an annual budget of about $750 million. He worked in OMB on the technology portions of the Administration’s 1990 budget and at DARPA to advance semiconductor manufacturing. He has extensive experience in measurement systems needed to support global trade and in developing government technology programs to stimulate the economy. He is a fellow of the IEEE.

Dr. Angelo L. Gattozzi joined the Center for Electromechanics at the University of Texas at Austin in 2000. His main focus has been on power modules for the electric gun program, resonant converters for high speed motors/generators with flywheel energy storage, design of special generators for energy harvesting from sea waves, and modeling the electric system with energy storage of the DDG51 class Navy destroyer and of some Army Tactical Operating Centers including renewable energy resources.