Received 21 November 2016; revised 7 March 2017 and 14 June 2017; accepted 23 July 2017.
Date of publication 28 July 2017; date of current version 16 September 2017.

Digital Object Identifier 10.1109/JPETS.2017.2732360

Fine-Grained Network Decomposition for Massively Parallel Electromagnetic Transient Simulation of Large Power Systems

ZHIYIN ZHOU (Student Member, IEEE) AND VENKATA DINAVAHI (Senior Member, IEEE)

Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada
CORRESPONDING AUTHOR: Z. ZHOU (zhiyin@ualberta.ca)

This work was supported by the Natural Science and Engineering Research Council of Canada.

ABSTRACT Electromagnetic transient (EMT) simulation is one of the most complex power system studies that requires detailed modeling of the study system including all frequency-dependent and nonlinear effects. Large-scale EMT simulation is becoming commonplace due to the increasing growth and interconnection of power grids, and the need to study the impact of system events of the wide area network. To cope with enormous computational burden, the massively parallel architecture of the graphics processing unit (GPU) is exploited in this paper for large-scale EMT simulation. A fine-grained network decomposition, called shattering network decomposition, is proposed to divide the power system network exploiting its topological and physical characteristics into linear and nonlinear networks, which adapt to the unique features of the GPU-based massive thread computing system. Large-scale systems, up to 240 000 nodes, with typical components, including synchronous machines, transformers, transmission lines, and nonlinear elements, and multiple levels modular multilevel converter with up to 6144 submodules, are tested and compared with mainstream simulation software to verify the accuracy and demonstrate the speed-up improvement with respect to sequential computation.

INDEX TERMS Electromagnetic transient (EMT) analysis, massively parallel, fine-grained network decomposition, nonlinear circuits, graphics processing unit (GPU), parallel numerical computing, power system simulation, parallel programming.

I. INTRODUCTION

Computer simulation plays an essential roles in modern power system design and analysis. The electromagnetic transient program (EMTP) [1], which analyzes the temporary electromagnetic phenomena in both off-line and real-time [2], such as changes of voltage, current and flux in a short time slice induced by switching, surges, faults, lightning strike or any other disturbances in the network [3], moreover, is also indispensable for the planning, construction and operation of realistic electrical generation, transmission and distribution facilities. Mainstream EMT tools, such as ATP, PSCAD/EMTDC®, EMTP-RV® and etc., developed over several decades, are routinely used in the power and energy industries, to study transient and dynamic phenomena over a wide frequency range [4]. These simulation tools comprise of extensive libraries of power system equipment models to accommodate most physical phenomena of practical significance. Due to the detailed, high-order and nonlinear models applied, the performance of mainstream EMT tools is bound by the sequential programing based on the CPU, running on a closely saturated clock frequency, when the network size becomes large. Meanwhile, the sparsity of the system matrix is increasing rapidly along with the system scale, which burdens the computational efficiency seriously. The traditional way to handle this is to create a large-scale sparse system matrix describing the physical electrical network, and then use a mathematical method for its solution [5].

The graphics processing unit (GPU), which brings supercomputing for the masses, has been described by many developers and researchers in various compute, analysis and simulation fields [6]–[8]. The application of high performance computing technology, such as multi-core CPU and many-core GPU, for solving large-scale power system problems, especially for dynamic and transient analysis is on the rise. There have already been contributions made in transient stability, dynamic state estimation, electromagnetic field and transient simulation, and power flow calculation areas [9]–[19]. For EMT simulation while there have been efforts to develop massively parallel models for linear passive elements and frequency-dependent elements, a comprehensive treatment of transients in nonlinear elements using a fully iterative solution for large systems is a desired and yet to be achieved objective. The single instruction multiple thread (SIMT), which is derived from single instruction multiple data (SIMD), provides the ability of branching, whereas the GPU has lightweight cores and coalescent memory. Therefore, the irregularity and unpredictability of the sparse construction seriously sabotages the SIMT execution and access efficiency of GPU cores compared with a multicore CPU-based sparse algorithm.

An alternative solution for SIMT execution is to partition the large system into similar small parts, which increases the rate of parallelism and the speed of convergence of solution for nonlinear systems using Newton iterations [20]. Exploiting the interface and boundary type sharing between subsystems, domain decomposition methods were proposed for parallel processing, such as Schur complement [21], additive Schwarz and multiplicative Schwarz method [22], which solve the system iteratively. Waveform relaxation methods, similarly, were developed for solving large sparse linear and nonlinear numerical systems iteratively [23]. In addition to the various pure mathematical methods, a large circuit can also be broken down into small parts based on its topology and physical characteristics. Diakoptics, introduced by Kron [24], showed that the method of system tearing can be a combination of equations and topology, and was further developed and systematized as a method to decompose electrical circuits [25], [26]. Taking into account the interconnection among subsystems, the system matrix exhibits a special bordered block diagonal pattern after transformation, which enabled parallelism.

This paper proposes a fine-grained decomposition method for sparse linear and nonlinear networks for large-scale EMT simulation implemented on a GPU-based parallel computing system. Considering the EMT simulation with detailed, nonlinear component models and the particular features of GPU-based computing, the large-scale sparse linear system, (which requires numerous random access and lowers data processing density) and the wide ranging nonlinear solution, (which is slow to converge and expensive to synchronize among threads) are still the main challenges of this work. Therefore, the partitioning method employed is much fine-grained than usual and is named *shattering decomposition* in this work. The compensation network decomposition method, evolving from diakoptics, is utilized to solve the partitioned linear system in parallel without iteration. The Jacobian domain decomposition method is proposed to decouple the nonlinear system to be solved by Newton-Raphson method iteratively.

The paper is organized as follows: Section II introduces the important characteristics of the computing system and electrical circuit for parallelism. Section III describes the simulation work flow and component modeling, where GPU-based parallelism can be engaged. Section IV gives the methods of shattering decomposition for linear and nonlinear systems. Section V explains the algorithms and implementations of the entire EMT simulation on a multi-GPU computing system. Finally, the experimental results for various large-scale test systems are shown, compared and analyzed in Section VI, followed by the conclusions in Section VII.

II. COMPUTER SYSTEM AND ELECTRICAL NETWORK

A. GPU-BASED COMPUTING SYSTEM

NVIDIA®'s GPU has already shown its power in parallel computing because of its native massive processing cores, high memory bandwidth and outstanding floating point capability [27]. Different from the heavyweight cores in multicore CPU whose threads are almost independent workers, the threads in SIMT GPU are numerous but lightweight. Thus, the performance of GPU-based computation depends to a great extent on the workload distribution and resource utilization. The typical architecture of CUDA [28] in Fig.1, shows hierarchies of thread and memory.

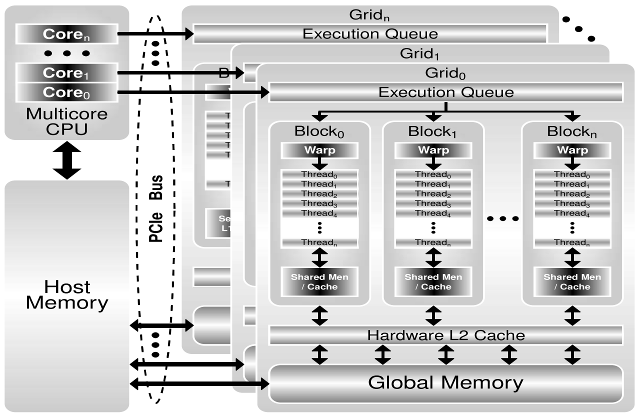


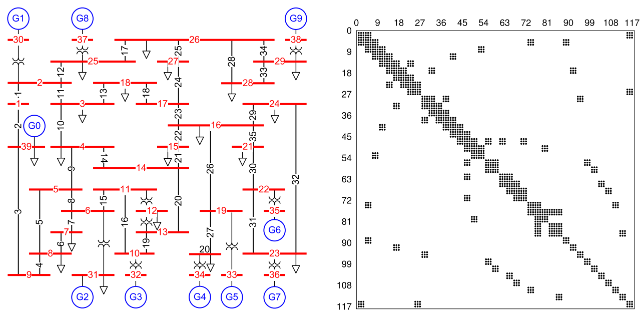
FIGURE 1. CUDA abstraction of device (GPU) and host (CPU).

Although the memories have high bandwidth, the channel, PCIe bus, between host and device is slow; thus avoiding those transfers unless they are absolutely necessary is vital for computational efficiency.

Based on the compute capability version 6.1 utilized in this work, each GPU device presents as a *grid*, in which there are up to 32 active *blocks* [27]. The *threads* in a block are grouped by warps. There are up to 4 active warps per block. Although a block maximally supports 1024 threads, only up to 32 threads in one warp can run simultaneously. In each block, there is 48KB of *shared memory* which is roughly 10x faster and has 100x lower latency than uncached *global memory*, whereas each thread has up to 255 registers running at the same speed as the cores. The overwhelming performance improvement was shown with avoiding and optimizing communication for parallel numerical linear algebra algorithms in various supercomputing platforms including GPU [29]. Making a full play of this critical resource can significantly increase the efficiency of computation, which requires the user to scale the problem perfectly and unroll the *for* loops in a particular way [30].

B. ELECTRICAL POWER NETWORK

The electrical power network transmitting energy from end to end is a sparse system, where each element (component) only links with few other elements (component) nearby. Fig. 2 shows the sparsity pattern of IEEE 39-bus power system, where each dot represents link between two nodes. There



(a) (b)

FIGURE 2. Sparsity of IEEE 39-bus power system. (a) 39-bus network. (b) Admittance matrix (*Y*).

are 117 nodes in total since all 39 buses are 3-phase. In order to avoid dealing with this sparsity during the computation, especially when the scale of network is considerably large, the fine-grained decomposition is applied during the simulation. Although the ideal target is to tear the system into component level pieces, such as bus-by-bus, for EMT simulation such a partition scheme would cause extra computing effort normally, such as data communication and connection networks.

The simulation performance relating to the scale of the subsystem has a step effect due to the warp execution of CUDA, which means it has the same performance within every 32-thread (1-warp) enlargement. Therefore, sparsity can be ignored in each warp, and the scale of the subsystems can be manipulated to meet the maximum size of the warp.

III. ELECTROMAGNETIC TRANSIENT MODELING

In modern electrical networks, the classical components include synchronous machines with control systems, transformers, transmission lines, and linear and nonlinear passive elements. Although the purpose of this work is to show the compute acceleration of fine-grained parallel EMT nonlinear simulation, detailed models are used to realize the computing power of the GPU. The basic theory of the electromagnetic transient program is to discretize the differential and integral equations in electrical circuits by Trapezoidal rule; and then to solve them repeatedly to find the numerical time-domain solutions, such as voltages and currents [4].

A. SYNCHRONOUS MACHINE WITH CONTROL SYSTEM

The universal machine model provides a unified mathematical framework to represent various types of rotating machines including synchronous, asynchronous and DC machine [31]. As shown in Fig. 3(a), the electrical part of the synchronous machine includes 3 stator armature windings $\{a, b, c\}$; one field winding f and up to 2 damper windings $\{D_1, D_2\}$ on the rotor direct d-axis; and up to 3 damper windings $\{Q_1, Q_2, Q_3\}$ on the rotor quadrature q-axis. The discretized

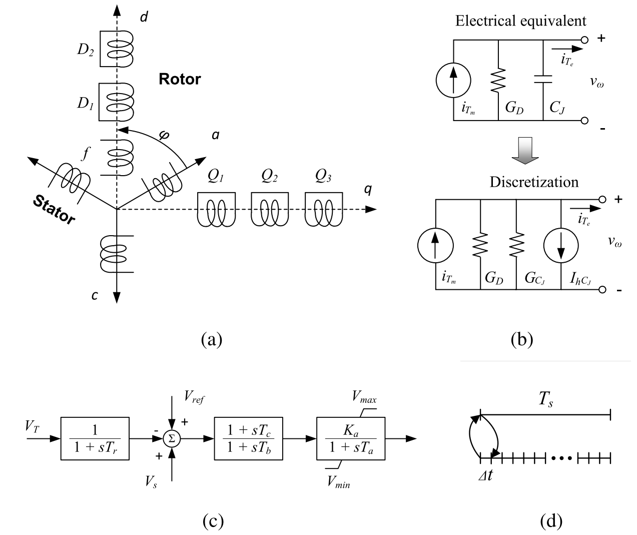


FIGURE 3. Universal machine model, control system, and time-step interface for synchronous machine. (a) Electrical side model. (b) Mechanical side model. (c) Excitation system. (d) Time-step interface.

winding equations after *dq0* conversion are described as

$$\boldsymbol{v}_{dq0}(t) = -\boldsymbol{R}\boldsymbol{i}_{dq0}(t) - \frac{2}{\Delta t}\boldsymbol{\lambda}_{dq0}(t) + \boldsymbol{u}(t) + \boldsymbol{V}_h \tag{1}$$

where $\boldsymbol{R}$ is the winding resistance, $\boldsymbol{\lambda}_{dq0}$ are the flux linkages, $\boldsymbol{u}$ are speed voltages and Δt is the simulation time-step. The mechanical part shown in Fig. 3(b) is represented as a linear electrical equivalent instead of traditional a mass-shaft system

$$i_{T_m} = C_J\frac{\mathrm{d}v_\omega}{\mathrm{d}t} + G_D v_\omega + i_{T_e} \tag{2}$$

where i_{T_m}, C_J, G_D and v_ω represent the equivalent mechanical torque, inertia, damping and rotor speed respectively [32]. The equivalent electromagnetic torque i_{T_e}, which interfaces to the electrical part, can be represented by the flux linkages $\boldsymbol{\lambda}_{dq0}$ and the machine currents $\boldsymbol{i}_{dq0}$ as

$$i_{T_e} = \lambda_d i_q - \lambda_q i_d. \tag{3}$$

Then the time discretization for linear passive elements is applied to (2), to obtain

$$v_\omega(t) = \frac{i_{T_m}(t) - i_{T_e}(t) - I_{hC_J}(t-\Delta t)}{G_D + G_{C_J}} \tag{4}$$

which merges the solution of mechanical part to that of the electrical part. The equivalent conductance G_{C_J} is given as

$$G_{C_J} = 2C_J/\Delta t. \tag{5}$$

B. CONTROL SYSTEM INTERFACE

Since the model of excitation control system, shown in Fig. 3(c), is different from that of the electrical network, whose equations are difficult to be merged into the nodal analysis equations in EMT simulation, they will be solved separately. On the other hand, the sample time (T_s), used in control system is much larger than the time step

$$T_s \gg \Delta t. \tag{6}$$

Therefore, there is an interface between the two subsystems, as shown in Fig. 3(d). After the machine network is solved, the solutions including voltages and currents are sent to the excitation system, and then its solutions are calculated, which are in turn sent back to the former in the next time-step of EMT computation. Since the excitation system appears as almost static to the EMT network, the error introduced by the time delay inserted is usually very small.

C. TRANSFORMER WITH MAGNETIC SATURATION

The n-winding transformer for EMT simulation with winding resistance $\boldsymbol{R}$ and leakage inductance $\boldsymbol{L}$ is represented as

$$\boldsymbol{v} = \boldsymbol{R}\boldsymbol{i} + \boldsymbol{L}\frac{\mathrm{d}\boldsymbol{i}}{\mathrm{d}t} \tag{7}$$

where $\boldsymbol{R}$ and $\boldsymbol{L}$ are $n \times n$ matrices; $\boldsymbol{v}$ and $\boldsymbol{i}$ are $n \times 1$ vectors of winding voltages and currents. The linear part of the transformer is modeled by admittance based model [33], and the saturation effect is represented by the extra nonlinear inductance connected to the secondary winding, which is joined to the linear inductance matrix, as shown in Fig. 4. By time discretization, (7) is rewritten as

$$\boldsymbol{i}(t) = \boldsymbol{G}_{eq}\boldsymbol{v}(t) + \boldsymbol{I}_h(t - \Delta t) \tag{8}$$

where the equivalent admittance is given as

$$\boldsymbol{G}_{eq} = \frac{\Delta t}{2}[\mathrm{I} + \frac{\Delta t}{2}\boldsymbol{L}^{-1}\boldsymbol{R}]^{-1}\boldsymbol{L}^{-1}. \tag{9}$$

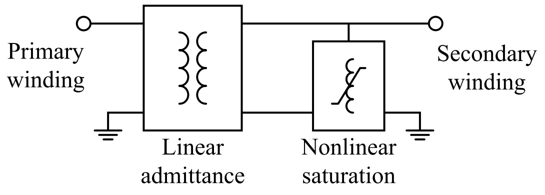


FIGURE 4. **Admittance-based transformer model with saturation.**

The nonlinear inductance taking care of the saturation effect contains the nonlinear relation between flux λ and branch current i, given as

$$g(\lambda, i) = 0. \tag{10}$$

On the other hand, the flux λ is the integral of node voltage v over a time-step

$$\lambda(t) = \lambda(t - \Delta t) + \int_{t-\Delta t}^{t} v(u)\mathrm{d}u. \tag{11}$$

Discretizing by Trapezoidal rule, we get

$$\lambda(t) = \frac{\Delta t}{2}v(t) + \Lambda_h. \tag{12}$$

Thus, the relation found by substituting λ in (10) with (12) between voltage and current of the nonlinear inductor is expressed as

$$g(\frac{\Delta t}{2}v(t) + \Lambda_h, i) = 0 \tag{13}$$

from which the Jacobian matrix can be derived by the derivative of voltage.

D. TRANSMISSION LINES

The universal line model (ULM) [34] is used for one of the most important components in electrical circuit, transmission line, which can represent both symmetrical and asymmetrical overhead lines and cables since it is constituted in the phase-domain directly. As shown in Fig. 5, the long line is modeled by two separate circuits with admittance $\boldsymbol{G}$ and history current source $\boldsymbol{I}_h$, sending end 'k' and receiving end 'm', linked by traveling waves. The KCL equation is expressed as

$$\boldsymbol{i}(t) = \boldsymbol{G}\boldsymbol{v}(t) - \boldsymbol{I}_h \tag{14}$$

where the history currents are given as

$$\boldsymbol{I}_h(t - \Delta t) = \boldsymbol{Y} * \boldsymbol{v}(t) - 2\boldsymbol{i}_i \tag{15}$$

and the incident currents $\boldsymbol{i}_i$ can be expressed by remote reflected currents $\boldsymbol{i}_r$ as

$$\boldsymbol{i}_i = \boldsymbol{H} * \boldsymbol{i}_r(t - \tau). \tag{16}$$

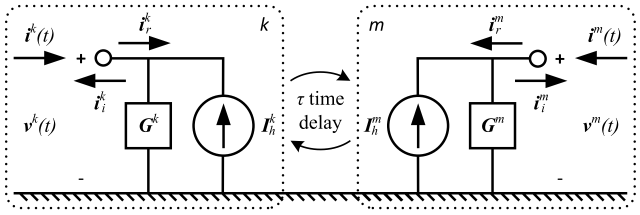


FIGURE 5. **Frequency-dependent transmission line model.**

In (15) and (16), the '$*$' denotes numerical matrix-vector convolution, which is a trade-off between the simplification of model complexity and computational resources; $\boldsymbol{Y}$ and $\boldsymbol{H}$ are the characteristic admittance and the propagation matrices respectively, which are approximated by finite-order rational functions using the vector fitting (VF) method [35]. From (16), there is a propagation time delay 'τ' between k and m ends. The admittance in (14) is given as

$$\boldsymbol{G} = (\frac{\Delta t}{2})/(1 - \boldsymbol{p}\frac{\Delta t}{2})\boldsymbol{r} + \boldsymbol{d} \tag{17}$$

where $\boldsymbol{r}, \boldsymbol{p}$ and $\boldsymbol{d}$ are the residues, poles and proportional terms of the vector fitting respectively.

E. LINEAR AND NONLINEAR PASSIVE ELEMENTS

As shown in Fig. 6(a), the linear passive elements, including resistor (R), inductor (L) and capacitor (C), can be represented as an equivalent admittance (G_{eq}), and a virtual current source (I_h)

$$i(t) = G_{eq}v(t) + I_h(t - \Delta t) \tag{18}$$

where I_h universally, defined as

$$I_h(t - \Delta t) = G_{eq}V_h(t - \Delta t). \tag{19}$$

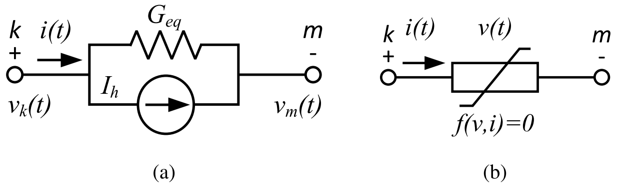


FIGURE 6. **Equivalent models for passive elements. (a) Linear. (b) Nonlinear.**

The equivalent resistances G_{eq} for L and C are given as

$$G_{eq}^L = \frac{\Delta t}{2L}, \quad G_{eq}^C = \frac{2C}{\Delta t}. \tag{20}$$

Besides linear passive elements, the nonlinear ones, shown in Fig. 6(b) also appear in the electrical network often, such as surge arrestor, nonlinear L and C, for which the voltage and current relation is defined by a nonlinear function

$$f(v, i) = 0 \tag{21}$$

instead of linear Ohm's law. Thus, a Newton type iteration method is involved to find their solutions. After the Jacobian matrix $\boldsymbol{J_v^F}$ of the nonlinear system $\boldsymbol{F}$ is updated by partial derivative of unknown voltages $\boldsymbol{v}$, the next approximate solutions $\boldsymbol{v}^{(n+1)}$ can be solved as

$$\boldsymbol{J_v^F}(\boldsymbol{v}^{(n)} - \boldsymbol{v}^{(n+1)}) = \boldsymbol{F}(\boldsymbol{v}^{(n)}) \tag{22}$$

where $\boldsymbol{v}^{(n)}$ are the current approximation. The iteration is repeated until $\boldsymbol{v}^{(n+1)}$ or $\boldsymbol{F}(\boldsymbol{v}^{(n+1)})$ is converged.

F. AC/DC CONVERTER

The structure of modular multilevel converter (MMC), which consists of a series of submodules (SMs), is shown in Fig. 7(a). The SM shown in Fig. 7(b) is made of two sets of IGBT-Diode units and paralleled with a capacitor, where the power electronic components can be modeled as functional switches [36] and the capacitor can be represented as an equivalent voltage source and a resistor as follows:

$$v_c\_h(t - \Delta t) = 2r_c i_c(t - \Delta t) - v_c\_h(t - 2\Delta t), \tag{23}$$

$$r_c = \frac{2\Delta t}{C}. \tag{24}$$

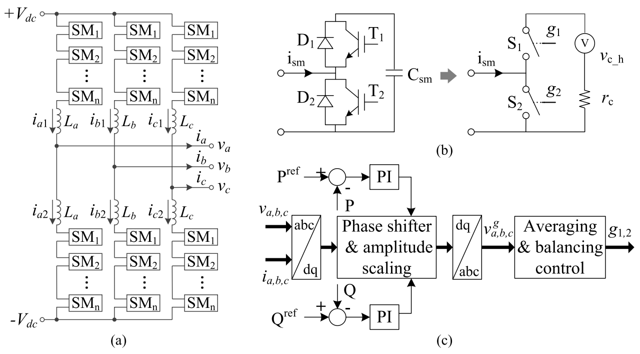


FIGURE 7. **AC/DC converter: (a) MMC structure, (b) switch model of SM, and (c) MMC control scheme.**

The control scheme, as shown in Fig. 7(c), includes the active and reactive power control, modulation signal phase shifter and amplitude scaling, and capacitor voltage averaging and balancing control [37]–[39]. The output gate signals, g_1 and g_2, are used to control the switches, S_1 and S_2, in the SM respectively. Since the gate signals are necessary for switching in each time-step, the sample rate for the control logic of MMC is the same with the time-step of the network solution.

IV. MULTILEVEL SHATTERING NETWORK DECOMPOSITION

According the characteristics of GPU-based computing system mentioned in Section II-A, the key point of the parallelism is to partition the large system into enough independent divisions such that the massive threads can access them fully, two levels of decomposition, coarse-grained and fine-grained, are proposed to decompose the large-scale electrical power circuit. As shown in Fig. 8, the original circuit is divided into *linear subsystems* (LS), *nonlinear subsystems* (NLS), and *control system* (CS) in the first-level decomposition by propagation delay based partitioning. In the second-level, the fine-grained decomposition methods are different for linear and nonlinear subsystems due to the solution methods. The linear subsystems are partitioned into *linear blocks* (LB) and *connecting network* (CN) by compensation network decomposition, through which the linear solutions can be obtained by solving the open-circuit voltages and compensation voltages in parallel. For nonlinear subsystems, the nonlinear components are detached from the network into *nonlinear blocks* (NLB) by Jacobian domain decomposition computed independently in parallel, and then the temporary results are interfaced to the rest of the connecting network to find the temporary solutions. Iterations of the above two-step computations are repeated until the solutions of the subsystems converge. All linear and nonlinear subsystems solutions are integrated to obtain the final solutions of the large-scale system. When the sample step arrives, the integrated solutions are sent to the *control block* (CB) to obtain the excitation signals which are fed back to the main circuit for the EMT computation of next time-step.

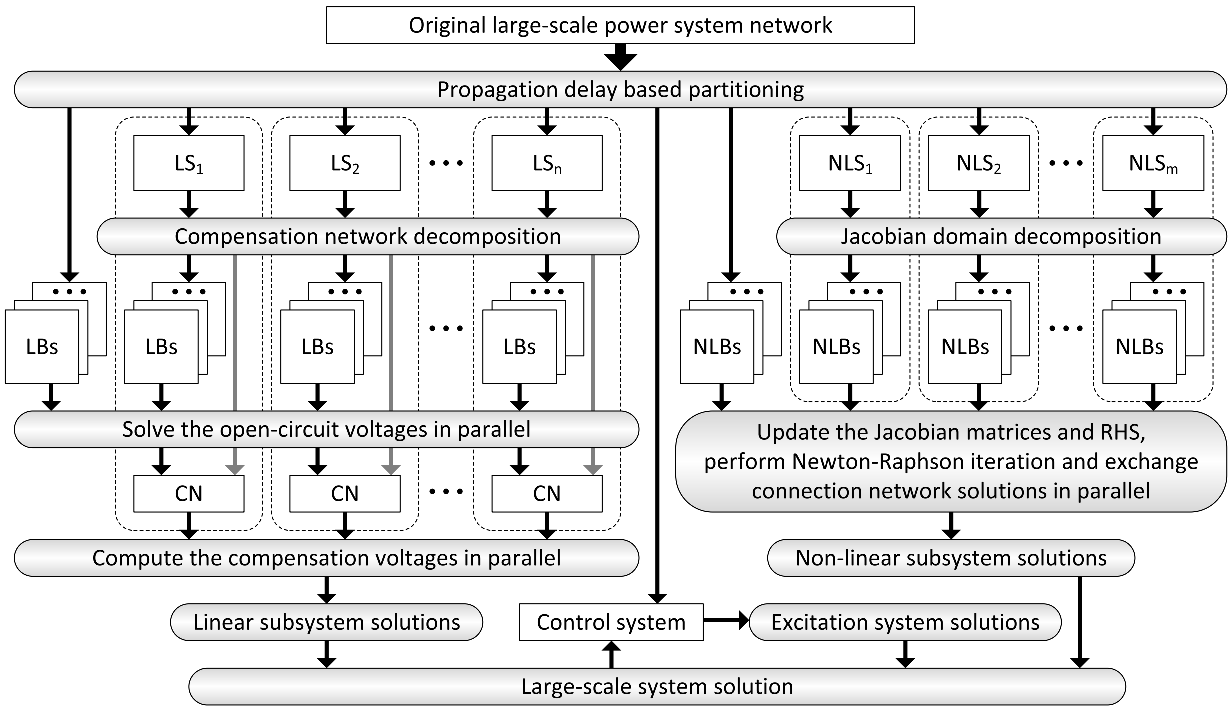

Original large-scale power system network
Propagation delay based partitioning
LS$_1$
LS$_2$
LS$_n$
NLS$_1$
NLS$_2$
NLS$_m$
Compensation network decomposition
Jacobian domain decomposition
LBs
NLBs
Solve the open-circuit voltages in parallel
Update the Jacobian matrices and RHS, perform Newton-Raphson iteration and exchange connection network solutions in parallel
CN
Compute the compensation voltages in parallel
Non-linear subsystem solutions
Linear subsystem solutions
Control system
Excitation system solutions
Large-scale system solution

FIGURE 8. **Shattering network decomposition.**

A. FIRST-LEVEL DECOMPOSITION (COARSE-GRAINED)

Observing the procedure of modeling and discretization, some components, such as transmission lines and control systems, have time delays inside or for connecting to external components, which provide the native characteristics to decouple the network based on propagation delay [40].

For example, in ULM of transmission line, the propagation delay τ in (16) is given as

$$\tau = (\beta/\omega)l = \sqrt{LC}l \tag{25}$$

where β is the phase constant, ω is the angular frequency, L is the intrinsic inductance, C is the intrinsic capacitance and l is the length of the transmission line. Similarly the control system can also be decoupled by Δt time delay. Since the transmission line is one of the essential components, the large network is partitioned into small subsystems, which are classified into linear subsystems decomposed into linear blocks and connecting networks, and nonlinear subsystems decomposed into nonlinear blocks. In the GPU architecture, shared memory is much faster than other memory types, so it is used to store the initial and intermediate data when the task is dispatched to the CUDA block, and only the result is sent back to global memory. On the other hand, registers of CUDA threads have more bandwidth than shared memory during arithmetic operations. The way to maximize the usage of registers, reaching the best performance, is to unroll the *for* loops in CUDA threads because the registers cannot be indexed and optimized by GPU automatically, which multiplies the workload of each CUDA thread to reduce communication, overlap memory access and increases occupation [41]. Since the general electrical network is 3-phase, the number of operations is multiples of 3 normally.

Considering the number of threads per CUDA warp, the amount of shared memory per CUDA block and the number of registers per CUDA thread, which are the fundamental criterions of determining the subsystem partition granularity, the node limitation of each network block is set to 12 (4×3-phase buses) and the unroll factors can be 2, 3 or their multiple. Therefore, the subsystems with less than 12 nodes can be sent to GPU directly as network blocks.

B. SECOND-LEVEL DECOMPOSITION (FINE-GRAINED)

In contrast, the subsystems with more than 4 buses will be divided into smaller blocks by fine-grained methods: compensation network decomposition for linear networks, and Jacobian domain decomposition for nonlinear networks respectively.

1) COMPENSATION NETWORK DECOMPOSITION

Considering the N-node sparse linear system (26) obtained using nodal analysis

$$\boldsymbol{Y}\boldsymbol{v} = \boldsymbol{i} \tag{26}$$

which can be rewritten as

$$\boldsymbol{Y}' \cdot \boldsymbol{v} = \boldsymbol{i} + \boldsymbol{i}'. \tag{27}$$

The skipped sparse elements of $\boldsymbol{Y}$, ignored in $\boldsymbol{Y}'$, represent the network components connecting between the blocks of subsystems. The injecting currents $\boldsymbol{i}'$ on RHS of (27) can be expressed by the B branch currents $\boldsymbol{I}$ of those components

$$\boldsymbol{i}' = \boldsymbol{c}\boldsymbol{I} \tag{28}$$

where the $N \times B$ matrix $\boldsymbol{c}$ indicates the connecting relations of the skipped components, whose element is just one among -1, 1 and 0 depending on the existence and direction of those currents. Similarly, the potential difference $\boldsymbol{E}$ across the connecting components can also be defined by the node voltages as

$$\boldsymbol{E} = \boldsymbol{d}\boldsymbol{v} \tag{29}$$

where the $B \times N$ matrix $\boldsymbol{d}$ is also a sign matrix. Inverting the $\boldsymbol{Y}'$ of (27), the solution $\boldsymbol{v}$ has two parts, open circuit solutions $\bar{\boldsymbol{v}}$ and compensated solutions $\boldsymbol{v}'$, defined as

$$\boldsymbol{v} = \bar{\boldsymbol{v}} + \boldsymbol{v}' = (\boldsymbol{Y}')^{-1}\boldsymbol{i} + (\boldsymbol{Y}')^{-1}\boldsymbol{i}'. \tag{30}$$

The open circuit solutions $\bar{\boldsymbol{v}}$ can be directly solved in parallel with

$$\bar{\boldsymbol{v}} = (\boldsymbol{Y}')^{-1}\boldsymbol{i} \tag{31}$$

since $\boldsymbol{Y}'$ is in block format. Applying Ohm's Law to the connecting components, we obtain

$$\boldsymbol{Z}\boldsymbol{I} = \boldsymbol{E} = \boldsymbol{d}\boldsymbol{v} = -\boldsymbol{c}^T\boldsymbol{v} \tag{32}$$

where $\boldsymbol{Z}$ is a B-order diagonal impedance matrix of the connecting components. Define an intermediate impedance matrix as

$$\boldsymbol{Z}' = \boldsymbol{Z} + \boldsymbol{c}^T(\boldsymbol{Y}')^{-1}\boldsymbol{c} \tag{33}$$

and then the compensation solutions $\boldsymbol{v}'$ can be obtained from

$$\boldsymbol{v}' = -(\boldsymbol{Y}')^{-1}\boldsymbol{c}(\boldsymbol{Z}')^{-1}\boldsymbol{c}^T\bar{\boldsymbol{v}}. \tag{34}$$

The compensation network decomposition partitions the subsystem into fine blocks (Fig. 9) and sparse admittance matrices are decoupled; therefore, the GPU-based dense direct linear solution algorithms, including matrix LU and inverse, can be implemented to solve the decomposed system in parallel without iterations.

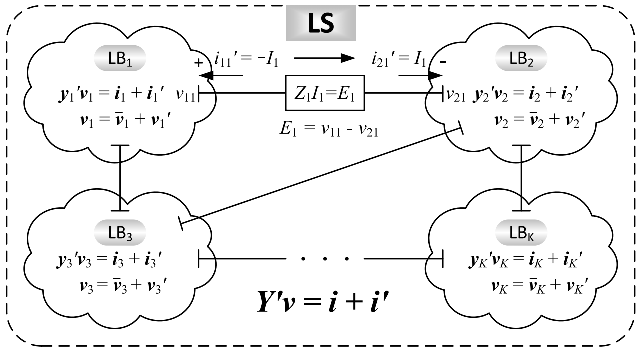


FIGURE 9. **Compensation network decomposition for LS.**

2) JACOBIAN DOMAIN DECOMPOSITION

For the solutions of nonlinear subsystems, Newton-Raphson (NR) iteration method is applied. If there is only a nonlinear component in the subsystem, NR will involve all components in it theoretically. However, the large sparse Jacobian matrix will impact the solution convergence and the application of parallelism. Therefore, Jacobian domain decomposition method is proposed to decouple the nonlinear elements into nonlinear blocks, as shown in Fig. 10.

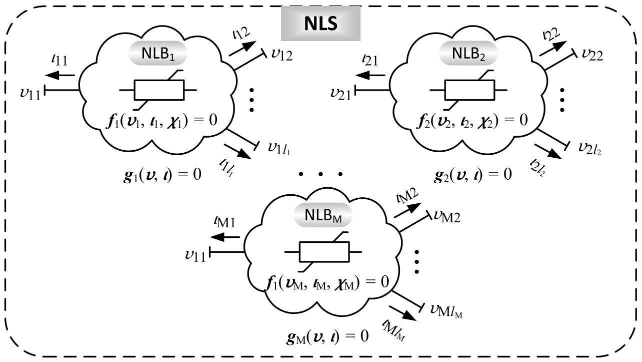


FIGURE 10. **Jacobian domain decomposition for NLS.**

A subsystem with M nonlinear components is decomposed into M nonlinear blocks. For the NLB$_k$ ($k = 1, 2, \cdots, M$), The nonlinear components can be expressed by a set of equations given as

$$\boldsymbol{f}_k(\boldsymbol{v}_k, \boldsymbol{\iota}_k, \boldsymbol{\chi}_k) = 0 \tag{35}$$

where $\boldsymbol{v}_k$ are the node voltages, $\boldsymbol{\iota}_k$ are currents connecting to the subsystem, and $\boldsymbol{\chi}_k$ are all internal variables of the kth block, which may be any internal voltages, currents, fluxes, etc. bonded with nonlinear relations. On the other hand, the linkages of the block in the subsystem can be described by a set of linear equations given as

$$\boldsymbol{g}_k(\boldsymbol{v}, \boldsymbol{\iota}) = 0 \tag{36}$$

where $\boldsymbol{v}$ are the set of connection node voltages of all nonlinear blocks of the subsystem, and $\boldsymbol{\iota}$ are connection node currents similarly. For the kth block, $\boldsymbol{v}$ and $\boldsymbol{\iota}$ can also be rewritten by $\boldsymbol{v}_k$, $\boldsymbol{\iota}_k$ and their complementary $\boldsymbol{v}_k^c$, $\boldsymbol{\iota}_k^c$. Since (36) is linear, $\boldsymbol{g}_k$ can be reformatted as $\bar{\boldsymbol{g}}_k$ for the node currents $\boldsymbol{\iota}_k$ represented as

$$\boldsymbol{\iota}_k = \bar{\boldsymbol{g}}_k(\boldsymbol{v}_k, \boldsymbol{v}_k^c, \boldsymbol{\iota}_k^c). \tag{37}$$

Substituting (37) into (35), the nonlinear equations are updated as

$$\bar{\boldsymbol{f}}_k(\boldsymbol{v}_k, \boldsymbol{v}_k^c, \boldsymbol{\iota}_k^c, \boldsymbol{\chi}_k) = 0 \tag{38}$$

which can be solved for $\boldsymbol{v}_k$ and $\boldsymbol{\chi}_k$ using NR method. The Jacobian matrix is given as

$$\boldsymbol{J}_k^{(n+1)} = \frac{d\bar{\boldsymbol{f}}_k(\boldsymbol{v}_k^{(n)}, \boldsymbol{v}_k^{c(n)}, \boldsymbol{\iota}_k^{c(n)}, \boldsymbol{\chi}_k^{(n)})}{d[\boldsymbol{v}_k\ \boldsymbol{\chi}_k]} \tag{39}$$

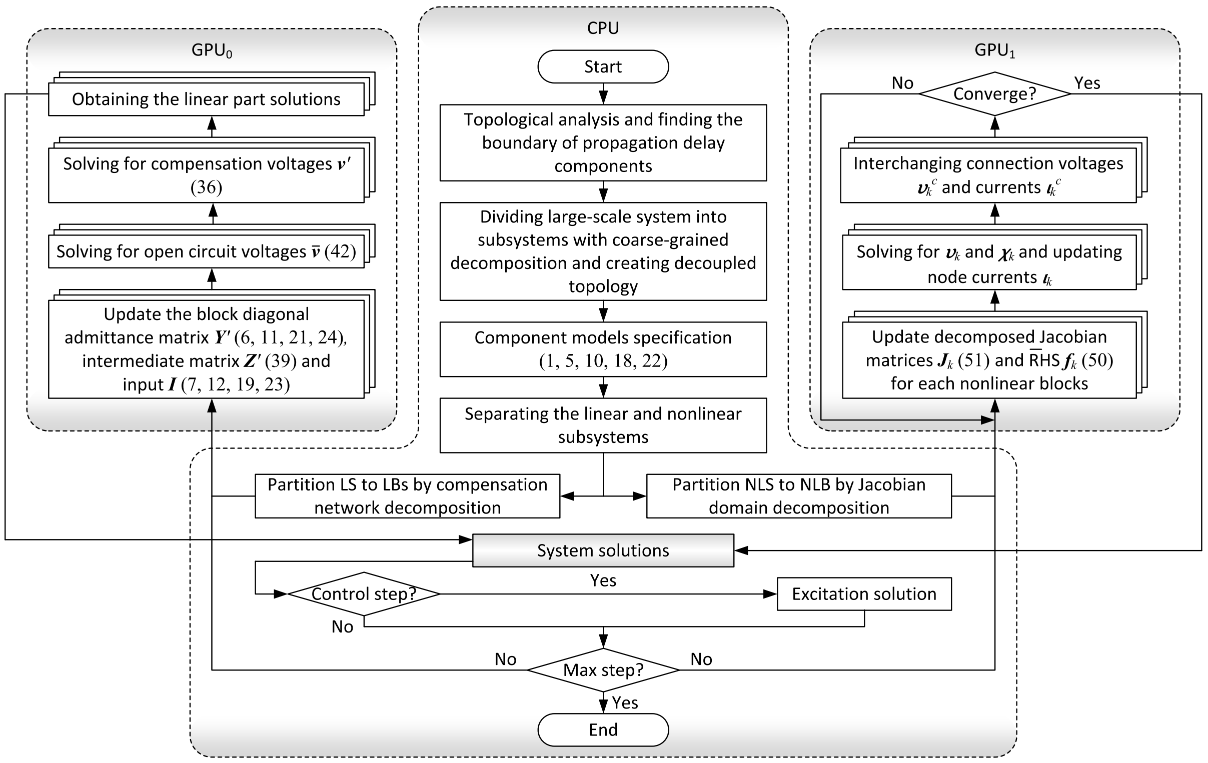

CPU
Start
Topological analysis and finding the boundary of propagation delay components
Dividing large-scale system into subsystems with coarse-grained decomposition and creating decoupled topology
Component models specification (1, 5, 10, 18, 22)
Separating the linear and nonlinear subsystems
GPU₀
Obtaining the linear part solutions
Solving for compensation voltages $\boldsymbol{v}'$ (36)
Solving for open circuit voltages $\bar{\boldsymbol{v}}$ (42)
Update the block diagonal admittance matrix $\boldsymbol{Y}'$ (6, 11, 21, 24), intermediate matrix $\boldsymbol{Z}'$ (39) and input $\boldsymbol{I}$ (7, 12, 19, 23)
GPU₁
Converge?
No
Yes
Interchanging connection voltages υ_k^c and currents ι_k^c
Solving for υ_k and χ_k and updating node currents ι_k
Update decomposed Jacobian matrices $\boldsymbol{J}_k$ (51) and RHS $\boldsymbol{f}_k$ (50) for each nonlinear blocks
Partition LS to LBs by compensation network decomposition
Partition NLS to NLB by Jacobian domain decomposition
System solutions
Control step?
Yes
No
Excitation solution
Max step?
No
No
Yes
End

FIGURE 11. **Fine-grained EMT simulation work flow.**

where $\boldsymbol{v}_k^c$ and $\boldsymbol{\iota}_k^c$ are the connection interfaces of other blocks inserted by the values of previous iteration. In this way, all nonlinear blocks are calculated independently and in parallel. When the new connection node voltages of all blocks are found, the node currents can be updated by (37), a part of which will be the connection voltages $\boldsymbol{v}^c$ and currents $\boldsymbol{\iota}^c$ for other blocks used in the next Jacobian matrix update. In contrast to the traditional NR method, the connection node voltages and currents of all blocks are interchanged with others to update the Jacobian matrix for the next iteration according to domain decomposition theory [42], [43]. The final solution for the subsystem can be found when all blocks have converged.

V. MASSIVELY PARALLEL IMPLEMENTATION ON MULTIPLE GPUs

The proposed fine-grained EMT simulation is implemented on a CPU-GPU heterogeneous platform. Considering the 64-bit double precision performance, which is used across the simulation, two Pascal architecture (GP104) NVIDIA® GPUs are mounted into the Intel® Xeon® E-2620 server with 32 GB memory running Windows 7 Enterprise 64-bit OS.

As shown in Fig. 11, the simulation starts with loading the netlist including the network connections and parameters, from which the topology of the network can be analyzed to find the boundaries of propagation delay for the first-level decomposition, such as transmission lines and control systems. The large network is then divided into subsystems by coarse-grained decomposition, and the bus node system is also rebuilt according to the new topology. After separating linear and nonlinear subsystems, they are partitioned into small linear blocks and nonlinear blocks with fine-grained decomposition methods as described in Section IV-B. The bus node numbers have to be remapped again to guarantee the admittance, and the resulting Jacobian matrices will be block diagonal. At this time, all the detailed component models including frequent-dependent line model are specified, the data structures on both host (CPU) and device (GPU) are determined and all necessary data are transfered from the host to the devices when the entire simulation process is branched. One GPU is responsible for linear blocks and the other takes charge of nonlinear blocks. Every component model listed in Section III is represented by a module, consisting a set of CUDA kernels, as well as solution methods, such as matrix operators, linear and nonlinear solvers [16]. According to the communication avoiding parallel theory in numerical linear algebra [44], in order to increase the register utilization inside each thread and minimize the data exchange between memories, the kernels are designed in small scale for limited register resource, the *for* loops are unrolled to make more data cached and individual thread work load is increased to extend the data lifetime inside the thread. Therefore, the per thread throughput is amplified, while the device occupancy per kernel is lowered, which can be compensated by kernel concurrent execution.

a: LINEAR SIDE

The component modules are applied to compose the admittance matrix $\boldsymbol{Y}'$ and initial inputs $\boldsymbol{i}$ in (27). Due to the independence of component modeling, all classified modules can run in parallel on the GPU. Since the optimized sparse admittance matrix $\boldsymbol{Y}'$ is decoupled, the open circuit linear solutions $\overline{\boldsymbol{v}}$ for all blocks run in parallel as well using (31). After all compensation voltages $\boldsymbol{v}'$ are solved by (34) at the same time, the solutions of linear blocks are found, which are then integrated to the solutions of the large system.

b: NONLINEAR SIDE

Since all nonlinear components are decoupled by Jacobian domain decomposition, the NR iterations are processed for all nonlinear blocks independently. The Jacobian matrices are composed by (39) with the updated nonlinear equations $\overline{\boldsymbol{f}}$ created during decomposition. When the next step voltages $\boldsymbol{v}_k^{(n+1)}$ for each blocks are solved, the node currents $\boldsymbol{\iota}_k^{(n+1)}$ can be obtained by the updated linkage functions $\overline{\boldsymbol{g}}$ in (37). They are interchanged among the blocks to update the connection voltages and currents for the next iteration. The parallel nonlinear solvers are synchronized when all solver loops are converged, and the results are sent to the host side as the other part of the system solution.

Combining the linear and nonlinear parts solutions, the large network system solutions for one time-step are found. Before approaching the next time-step, the synchronization of control system and EMT network is checked on the CPU, and if 'Yes', the control system solution will be calculated for the next EMT time-step. In order to parallelize the tasks with various algorithms that cannot be contained in a same kernel with different blocks, and cover the data transfer time between host and device, multiple streams are used to group independent kernels. As shown in Fig. 12, the dependent kernels, such as a set of kernels for a component module, are assigned to the same stream, which are executed in serial, while the kernels in different streams are independent without any data or procedure interference. Firstly, the data for each

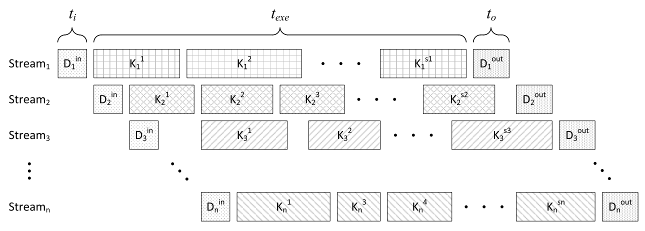


FIGURE 12. **Concurrent execution with multiple streams.**

stream are copied from host to device costing t_i; then the set of kernels belonging to the stream are executed in t_{exe}; lastly, the results of the stream execution are copied back to host consuming t_o. The data transfer cost can be effectively covered only if $t_{exe} > (t_i + t_o)$, which is scheduled delicately. In addition, the execution of streams can also be concurrent when the GPU hardware still has enough resources available, which increases the overall occupation of GPU since the kernel are designed with low occupancy.

VI. SIMULATION CASE STUDIES

In order to show the accuracy of transients, and the acceleration for the proposed simulator, three test cases are utilized.

In the first test case, various transient behaviors are presented, and the simulation results are validated by the EMT software ATP and EMTP-RV®. In the second test case, the accelerating performance of GPU, whose execution times on various system scales are compared to those EMTP-RV®, is shown and analyzed by running the EMT simulation on the extended large-scale power systems.

In the last test case, the 3-phase modular multilevel converter (MMC) based AC/DC converter is simulated and compared with different submodule levels. The hardware and software environment of the test system is listed in Table 1, and the parameters for the test cases are given in the Appendix.

TABLE 1. **Test system specification.**

CPU	Intel Xeon E5-2620
Main memory	32GB
GPU	GeForce GTX 1080 (Pascal) $\times$ 2
Video memory	8GB $\times$ 2 (16GB)
OS	Windows 7 Enterprise, 64-bit

A. CASE STUDY A

The synchronous machine (SM), two transformers (T$_1$, T$_2$) and the arrester (MOV) are nonlinear components in the test system, as shown in Fig. 13. The first switch (SW$_1$) closes at

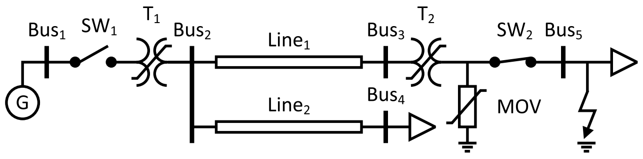


FIGURE 13. **Single-line diagram for Case Study A.**

0.01s, the ground fault happens at 0.15s, and then the second switch (SW$_2$) opens at 0.19s to clear the fault. The total simulation time is 0.3s with $20\mu s$ time-step.

The 3-phase voltages at Bus$_2$ and Bus$_3$, currents through Bus$_2$, power angle, electromagnetic torque, active power and reactive power are shown in Fig. 14, where both voltage and current waves illustrate good agreement with ATP and EMTP-RV®.

B. CASE STUDY B

In order to show the acceleration of GPU based EMT simulation, large-scale power systems are built, which are based on the IEEE 39-bus network as shown in Fig. 2(a).

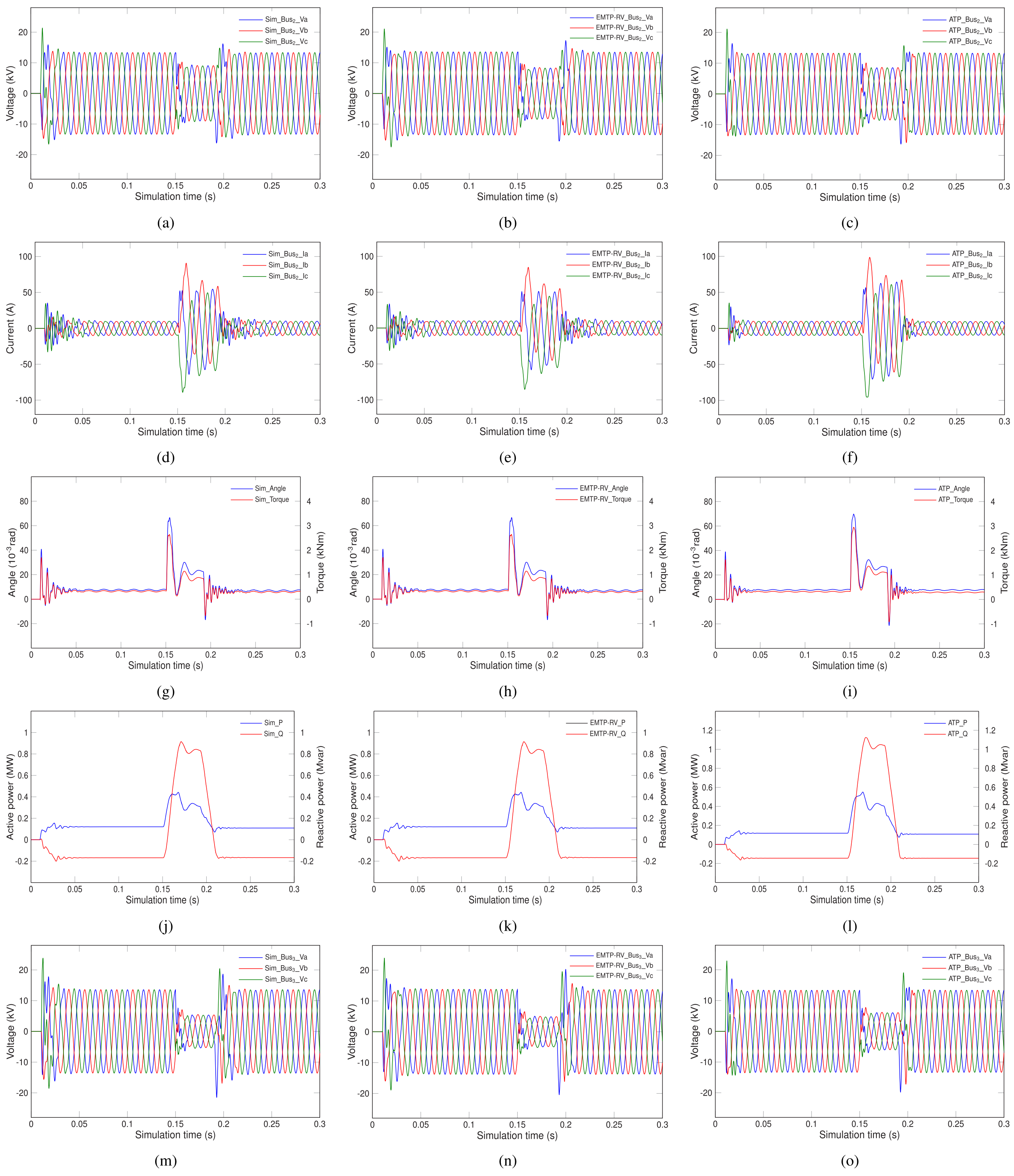


FIGURE 14. Simulation results of Case Study A, including three-phase Bus voltages, branch currents, angles, torques, active power, and reactive power, compared among GPU-based simulation, EMTP-RV®, and ATP. (a) Bus$_2$ voltages from GPU-based simulation. (b) Bus$_2$ voltages from EMTP-RV®. (c) Bus$_2$ voltages from ATP. (d) Bus$_2$ currents from GPU-based simulation. (e) Bus$_2$ currents from EMTP-RV®. (f) Bus$_2$ currents from ATP. (g) Angle and torque from GPU-based simulation. (h) Angle and torque from EMTP-RV®. (i) Angle and torque from ATP. (j) P and Q from GPU-based simulation. (k) P and Q from EMTP-RV®. (l) P and Q from ATP. (m) Bus$_3$ voltages from GPU-based simulation. (n) Bus$_3$ voltages from EMTP-RV®. (o) Bus$_3$ voltages from ATP.

Considering the interconnection is a path of power grid growth, the large-scale networks are obtained by duplicating the Scale 1 system and interconnecting by transmission lines; nevertheless, the proposed shattering decomposition method is applicable to general large-scale power systems. As shown in Table 2, the test systems are extended up to 3×79872 (239616) buses. All networks are decomposed into

TABLE 2. Comparison of execution time for various networks among CPU, single-GPU, and multi-GPU, for simulation duration 100*ms* with time-step 20μs.

Scale	3ϕ buses	Blocks			Execution time (s)				Speedup		
		LBs	NLBs	CBs	EMTP-RV®	CPU	1-GPU	2-GPU	CPU	1-GPU	2-GPU
1	39	28	21	10	1.24	1.38	1.16	0.93	0.89	1.06	1.33
2	78	56	42	20	2.71	2.87	1.52	1.36	0.94	1.78	1.99
4	156	112	84	40	5.54	5.53	2.03	1.77	1.00	2.73	3.14
8	312	224	168	80	11.94	11.22	2.96	2.37	1.06	4.04	5.03
16	624	448	336	160	26.50	23.12	4.63	3.35	1.15	5.73	7.91
32	1248	896	672	320	60.65	48.16	7.98	5.19	1.26	7.60	11.68
64	2496	1792	1344	640	142.31	100.11	14.42	8.72	1.42	9.87	16.31
128	4992	3584	2688	1280	323.26	210.76	28.36	15.83	1.53	11.40	20.42
256	9984	7168	5376	2560	705.92	439.26	55.47	30.23	1.61	12.73	23.35
512	19968	14336	10752	5120	1513.35	892.45	109.29	57.78	1.70	13.85	26.19
1024	39936	28672	21504	10240	3314.24	1863.66	225.17	116.86	1.78	14.72	28.36
2048	79872	57344	43008	20480	7033.61	3796.37	454.48	234.37	1.85	15.48	30.01

LBs, NLBs and CBs after fine-grained decomposition in the unified patterns. For instance, the 39-bus network is divided into 28 LBs, 21 NLBs and 10 CBs. The simulation is based on CPU, 1-GPU and 2-GPU computational systems from 0 to 100ms with 20μs time-step respectively, using double precision and 64-bit operation system. All test cases are extended sufficiently long to suppress the deviation of the software timer, which starts after reading the circuit net-list and parameters, including network decomposition, memory copy, component model calculation, linear/nonlinear solution, node voltage/current update, result output and transmission delay.

The scaled test networks are given in Table 2, including network size, bus number and partition. The execution time for each network is listed in order of network size and categorized by the type of computing systems as well as the speedup referred to the performance on CPU. In the plotted Fig. 15 along the 1-GPU speedup curve, the speedup increases slowly when network size is small (lower than 4 scales) since the GPU cannot be fed enough workload; for the network scale from 4 to 32, the acceleration climbs fast, showing the computational power of the GPU is released by fetching a greater amount of data; when the network size is more than 32, the performance approaches a constant since the computational capability of GPU closes to saturation. In the case of 2-GPU system, the trend of speedup increase is similar to the 1-GPU case except that the saturation point is put off because of the doubled computational capability.

Due to the nonlinear relationship of the execution time to the system scale, the bar diagrams of execution times for various system scales are zoomed using a logarithmic log axis to obtain a detailed view. Additionally, it can be noticed that the performance of CPU is also enhanced by the proposed decomposition method since the divided circuit

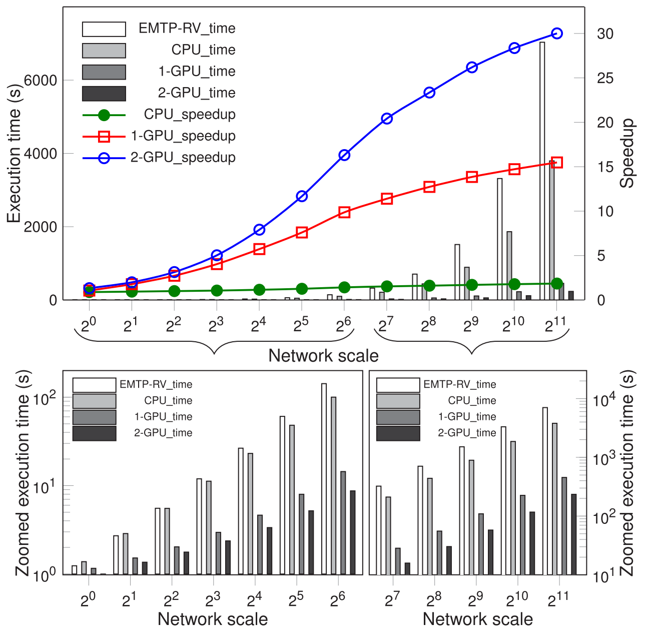


FIGURE 15. Execution time and speedup for varying scales of test networks on CPU, one GPU, and two GPUs-based programs compared to EMTP-RV®.

blocks simplify the sparse data structure to dense one along with the increasing system scale, thus the systems can be solved by dense solver and avoid the extra cost of dealing with the sparse structure, such as nonzero elements analysis, which is involved in every solution. In that case, the computation load is almost linearly related to the system scale comparing with the nonlinear traditional sparse solver. Owing to the shattering network decomposition, the computation load can be well-distributed to each compute device so that the overall performance of a computing system is decided by the number of processors, following the Gustafson-Barsis' law [46]. The average execution time for one time-step is listed in Table 3 due to the different convergent speed of each time-step.

TABLE 3. Average execution time (*ms*) for one time-step.

Scale	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}
EMTP-RV®	0.25	0.54	1.11	2.39	5.30	12.13	28.46	64.65	141.18	302.67	662.85	1406.72
CPU	0.28	0.57	1.11	2.24	4.62	9.63	20.02	42.15	87.85	178.49	372.73	759.27
1-GPU	0.23	0.30	0.41	0.59	0.93	1.60	2.88	5.67	11.09	21.86	45.03	90.90
2-GPU	0.19	0.27	0.35	0.47	0.67	1.04	1.74	3.17	6.05	11.56	23.37	46.87

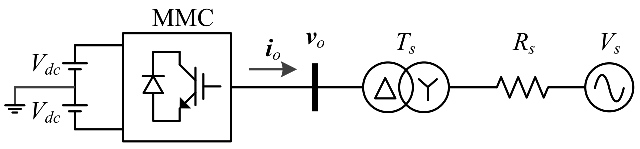


FIGURE 16. Single-line diagram for Case Study C.

When the network scale is up to 2^{11} (2048), which is close to the memory limitation of the computing system, the 2-GPU

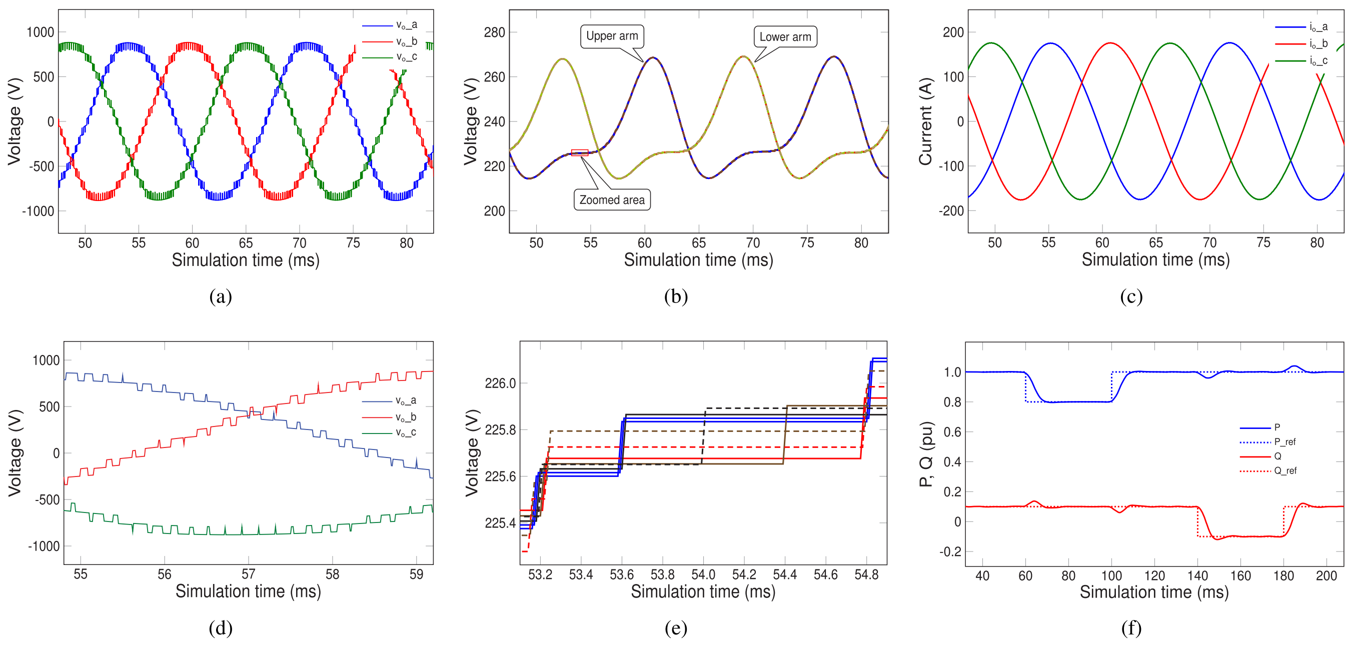


FIGURE 17. Simulation results of Case Study C, including three-phase output voltages, SM capacitor voltages, output currents, power, and reactive power, from GPU-based simulation with 8-SM per arm (17-level) MMC. (a) Three-phase output voltages of 17-level MMC from GPU-based simultion. (b) SM capacitor voltages of 17-level MMC from GPU-based simultion. (c) Three-phase output currents of 17-level MMC from GPU-based simultion. (d) Zoomed output voltages of MMC. (e) Zoomed SM capacitor voltages of MMC. (f) Active and reactive power control.

TABLE 4. CPU and GPU execution times of three-phase AC/DC converter for 0.5s duration with 10μs time-step.

N_{SM}	N_{SM}	Execution time (s)			Speedup	
per arm	total	CPU	1-GPU	2-GPU	1-GPU	2-GPU
8	48	9.77	1.98	1.91	4.93	5.12
16	72	17.75	2.18	2.02	8.14	8.79
32	144	34.83	2.61	2.27	13.34	15.34
64	288	68.89	4.16	3.12	17.30	22.08
128	768	131.72	5.92	4.87	23.03	27.05
256	1536	254.95	8.64	6.96	29.51	36.63
512	3072	485.58	12.87	9.55	37.73	50.85
1024	6144	902.69	17.53	14.03	51.49	64.34

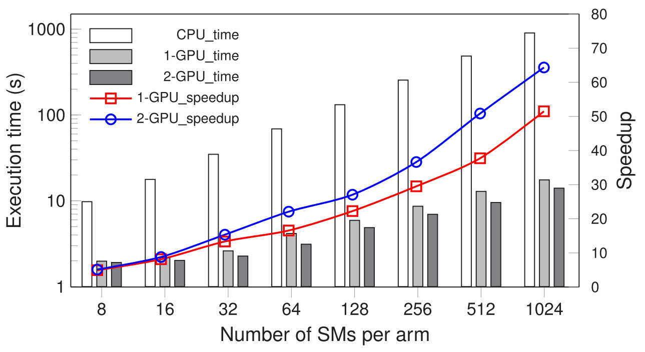


FIGURE 18. Execution time and speedup of simulation for varying levels of MMC on CPU, one GPU, and two GPUs-based programs.

system doubles the performance of the 1-GPU system and attains 30 times faster than EMTP-RV®.

C. CASE STUDY C

The AC/DC converter based on MMC, as shown in Fig. 16, is used to evaluate the power electronic type of switching in GPU-based EMT simulation. Due to the proposed fine-grained decomposition algorithm, all 6 arms in 3-phase MMC are decoupled and each SM in one arm is processed by one thread. The waveforms in Fig. 17 show the EMT simulation results of the converter with 8 SMs per arm (17-level) MMC with 10μs time-step. The 3-phase output voltages of MMC are shown in Fig. 17(a) and zoomed in Fig. 17(d) between 56ms to 59ms; the capacitor voltages of upper and lower arms of SMs in MMC are shown in Fig. 17(b) and the waveforms inside the marked area on upper arm curves are zoomed in Fig. 17(e) between 53.2ms and 54.8ms; 3-phase output currents are shown in Fig. 17(c); active and reactive power control results are shown in Fig. 17(f), which correctly follow the reference P and Q signals.

The performance of GPU-based massively parallel EMT algorithm with the proposed fine-grained decomposition is compared to CPU-based simulation by varying the number of SMs per arm in MMC. The execution times from CPU, 1-GPU and 2-GPU based simulation of 3-phase MMC converter with a 10μs time-step during 0.5s simulation are listed and compared in Table 4 from 8 SMs per arm (17-level) to 1024 SMs per arm (2049-level) in MMC. In Fig. 18, the bar graph shows the comparison of execution among various computational platforms and curves illustrate that the speedup keeps increasing along with the number of SMs in

MMC, which is close to 51 times for 1-GPU platform and reaches 64 times for 2-GPU platform compared to the single thread CPU simulation. It is obvious that the execution time is almost doubled when the number of SMs in MMC is doubled on CPU-based simulation; however, it grows much slower for GPU-based simulation. Since the increase of speedup is close to linear, the computational complexity order of the EMT simulation is reduced effectively by the massively parallel computation on the GPUs.

VII. CONCLUSION

Electromagnetic transient simulation of large-scale nonlinear power grids is computationally demanding, and it is therefore imperative to accelerate the simulation which is used to conduct a wide variety of studies by electric utilities. The shattering network decomposition methods proposed in the paper

partitions the power electrical circuit, decouples the linear and nonlinear solution, and accelerates the power electronic converter circuit simulation working with high frequency switching, which enables the transient simulation to take advantage of massively parallel computing platform conveniently and unleashes the computational power of GPUs. All the component models employed are detailed and the solution is fully iterative. Along with the increasing scale, fully decomposed simulation can be easily deployed on to multi-GPU computing system and implement the massively parallel algorithms, so that the maximum performance of simulation can be obtained according Gustafson-Barsis' law.

APPENDIX A

The parameters for Case Study A are as follows:

1) Synchronous machine parameters: 10MVA, 3.5kV, Y-connected, field current: 5A, 2 poles, 60Hz, moment of inertia: 4Mkg·m^2/rad and damping: 50kg·m/s/rad.
2) Transmission line parameters: Line$_1$: three conductors, resistance: 0.0583/km, diameter: 3.105cm, line length: 50km; Line$_2$: three conductors, resistance: 0.0583/km, diameter: 3.105cm, line length: 150km. Line geometry: flat horizontal phase spacing; horizontal distance between adjacent phases = 4.87m; vertical distance: phases a to ground, c to ground = 30m, phase b to ground = 28m, and shield wire to tower arm = 6m.
3) Transformer parameters: T$_1$: 10MVA, 3.5kV/22kV, $X_{\text{leakage}} = 6.89e^{-3}$pu, Y-Y connection; T$_2$: 10MVA, 22kV/3.5kV, $X_{\text{leakage}} = 0.192$pu and Y-Y connection.
4) Arrester parameters: V_{ref} = 5kV, multiplier(p) = 1.1 and exponent(q) = 26.
5) Loads parameters: R = 1kΩ and L = 1mH.

The parameters for the Scale-1 39-bus test system of Case Study B are as follows:

1) Synchronous machine parameters: 1000MVA, 22kV, Y-connected, field current: 2494A, 2 poles, 60Hz, moment of inertia: 4Mkg·m^2/rad and damping: 6780kg·m/s/rad.
2) ULM transmission line parameters: Line 1 - 35: three conductors, resistance: 0.0583/km, diameter: 3.105cm, line length: 50km (Line 5, 6, 7, 8, 15, 16, 18, 19, 23, 27, 29, 30, 31, 35), 150km (Line 2, 3, 4, 9, 10, 11, 13, 14, 20, 21, 22, 24, 25, 26, 32, 33) and 500km (Line 1, 12, 17, 28, 34). Line geometry: flat horizontal phase spacing; horizontal distance between adjacent phases = 4.87m; vertical distance: phases a to ground, c to ground = 30m, phase b to ground = 28m, and shield wire to tower arm = 6m.
3) Transformer parameters: 1000MVA, 22V/220kV, $X_{\text{leakage}} = 9.24e^{-3}$pu and Y-Y connection;
4) Loads parameters: R = 1kΩ and L = 1mH.

The parameters for MMC of Case Study C are as follows:

1) MMC parameters: DC voltage V_{dc} = 1kV, arm inductance L_a=L_b=L_c = 150mH, SM capacitance C_m = 4mF, carrier frequency f_c = 2500Hz, system frequency f_s = 60Hz.
2) AC Source parameters: transformer reactance X_T = 1.88Ω, V_s = 1kV, R_s = 5Ω.

REFERENCES

[1] H. W. Dommel, "Digital computer solution of electromagnetic transients in single and multiphase networks," *IEEE Trans. Power App. Syst.*, vol. PAS-88, no. 4, pp. 388–399, Apr. 1969.

[2] M. D. O. Faruque *et al.*, "Real-time simulation technologies for power systems design, testing, and analysis," *IEEE Power Energy Technol. Syst. J.*, vol. 2, no. 2, pp. 63–73, Jun. 2015.

[3] H. W. Dommel and W. S. Meyer, "Computation of electromagnetic transients," *Proc. IEEE*, vol. 62, no. 7, pp. 983–993, Jul. 1974.

[4] H. W. Dommel, *EMTP Theory Book*. Portland, OR, USA: Bonneville Power Admin., 1984.

[5] J. W. Demmel, J. R. Gilbert, and X. S. Li, "An asynchronous parallel supernodal algorithm for sparse Gaussian elimination," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 4, pp. 915–952, 1999.

[6] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym, "NVIDIA Tesla: A unified graphics and computing architecture," *IEEE Micro*, vol. 28, no. 2, pp. 39–55, Mar./Apr. 2008.

[7] D. Blythe, "Rise of the graphics processor," *Proc. IEEE*, vol. 96, no. 5, pp. 761–778, May 2008.

[8] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879–899, May 2008.

[9] A. Gopal, D. Niebur, and S. Venkatasubramanian, "DC power flow based contingency analysis using graphics processing units," in *Proc. IEEE Lausanne Power Tech*, Jul. 2007, pp. 731–736.

[10] V. Jalili-Marandi and V. Dinavahi, "Large-scale transient stability simulation on graphics processing units," in *Proc. IEEE Power Energy Soc. General Meet.*, Calgary, AB, USA, Jul. 2009, pp. 1–6.

[11] N. Garcia, "Parallel power flow solutions using a biconjugate gradient algorithm and a Newton method: A GPU-based approach," in *Proc. IEEE PES General Meet.*, Minneapolis, MN, USA, Jul. 2010, pp. 1–4.

[12] V. Jalili-Marandi and V. Dinavahi, "SIMD-based large-scale transient stability simulation on the graphics processing unit," *IEEE Trans. Power Syst.*, vol. 25, no. 3, pp. 1589–1599, Aug. 2010.

[13] V. Jalili-Marandi, Z. Zhou, and V. Dinavahi, "Large-scale transient stability simulation of electrical power systems on parallel GPUs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1255–1266, Jul. 2012.

[14] H. Karimipour and V. Dinavahi, "Extended Kalman filter-based parallel dynamic state estimation," *IEEE Trans. Smart Grid*, vol. 6, no. 3, pp. 1539–1549, May 2015.

[15] H. Karimipour and V. Dinavahi, "Parallel relaxation-based joint dynamic state estimation of large-scale power systems," *IET Generat., Transmiss. Distrib.*, vol. 10, no. 2, pp. 452–459, 2016.

[16] Z. Zhou and V. Dinavahi, "Parallel massive-thread electromagnetic transient simulation on GPU," *IEEE Trans. Power Del.*, vol. 29, no. 3, pp. 1045–1053, Jun. 2014.

[17] J. K. Debnath, A. M. Gole, and W. K. Fung, “Graphics-processing-unit-based acceleration of electromagnetic transients simulation,” *IEEE Trans. Power Del.*, vol. 31, no. 5, pp. 2036–2044, Oct. 2016.

[18] P. Liu and V. Dinavahi, “Finite-difference relaxation for parallel computation of ionized field of HVDC lines,” *IEEE Trans. Power Del.*, to be published.

[19] V. Roberge, M. Tarbouchi, and F. Okou, “Parallel power flow on graphics processing units for concurrent evaluation of many networks,” *IEEE Trans. Smart Grid*, vol. 8, no. 4, pp. 1639–1648, Jul. 2015.

[20] N. Fröhlich, B. M. Riess, U. A. Wever, and Q. Zheng, “A new approach for parallel simulation of VLSI circuits on a transistor level,” *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 45, no. 6, pp. 601–613, Jun. 1998.

[21] P. Aristidou, D. Fabozzi, and T. Van Cutsem, “Dynamic simulation of large-scale power systems using a parallel Schur-complement-based decomposition method,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 10, pp. 2561–2570, Oct. 2014.

[22] H. Karimipour and V. Dinavahi, “Parallel domain-decomposition-based distributed state estimation for large-scale power systems,” *IEEE Trans. Ind. Appl.*, vol. 52, no. 2, pp. 1265–1269, Mar./Apr. 2016.

[23] Y. Liu and Q. Jiang, “Two-stage parallel waveform relaxation method for large-scale power system transient stability simulation,” *IEEE Trans. Power Syst.*, vol. 31, no. 1, pp. 153–162, Jan. 2016.

[24] G. Kron, *Diakoptics; Piecewise Solution of Large-Scale Systems*. London, U.K.: MacDonald, 1963.

[25] H. H. Happ, “Diakoptics and piecewise methods,” *IEEE Trans. Power App. Syst.*, vol. PAS-89, no. 7, pp. 1373–1382, Sep. 1970.

[26] D. Montenegro, G. A. Ramos, and S. Bacha, “Multilevel a-diakoptics for the dynamic power-flow simulation of hybrid power distribution systems,” *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 267–276, Feb. 2016.

[27] “NVIDIA Tesla P100: The most advanced datacenter accelerator ever built—Featuring Pascal GP100, the world’s fastest GPU,” NVIDIA Corp., Santa Clara, CA, USA, White Paper P100, 2016.

[28] *CUDA C Programming Guide*, NVIDIAr Corp., Santa Clara, CA, USA, Sep. 2015.

[29] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz, “Minimizing communication in linear algebra,” *SIAM J. Matrix Anal. Appl.*, vol. 32, no. 3, pp. 866–901, Sep. 2011.

[30] V. Volkov and J. W. Demmel, “Benchmarking GPUs to tune dense linear algebra,” in *Proc. ACM/IEEE Conf. Supercomput.*, Austin, TX, USA, Nov. 2008, pp. 1–11.

[31] H. K. Lauw and W. S. Meyer, “Universal machine modeling for the representation of rotating electric machinery in an electromagnetic transients program,” *IEEE Trans. Power App. Syst.*, vol. 101, no. 6, pp. 1342–1350, Jun. 1982.

[32] H. K. Lauw, “Interfacing for universal multi-machine system modeling in an electromagnetic transients program,” *IEEE Trans. Power App. Syst.*, vol. 104, no. 9, pp. 2367–2373, Sep. 1985.

[33] V. Brandwajn, H. W. Dommel, and I. Dommel, “Matrix representation of three-phase n-winding transformers for steady-state and transient studies,” *IEEE Trans. Power App. Syst.*, vol. PAS-101, no. 6, pp. 1369–1378, Jun. 1982.

[34] A. Morched, B. Gustavsen, and M. Tartibi, “A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables,” *IEEE Trans. Power Del.*, vol. 14, no. 3, pp. 1032–1038, Jul. 1999.

[35] B. Gustavsen and A. Semlyen, “Simulation of transmission line transients using vector fitting and modal decomposition,” *IEEE Trans. Power Del.*, vol. 13, no. 2, pp. 605–614, Apr. 1998.

[36] A. Myaing, M. O. Faruque, V. Dinavahi, and C. Dufour, “Comparison of insulated gate bipolar transistor models for FPGA-based real-time simulation of electric drives and application guideline,” *IET Power Electron.*, vol. 5, no. 3, pp. 293–303, Mar. 2012.

[37] S. Debnath, J. Qin, B. Bahrani, M. Saeedifard, and P. Barbosa, “Operation, control, and applications of the modular multilevel converter: A review,” *IEEE Trans. Power Electron.*, vol. 30, no. 1, pp. 37–53, Jan. 2015.

[38] H. Saad *et al.*, “Dynamic averaged and simplified models for MMC-based HVDC transmission systems,” *IEEE Trans. Power Del.*, vol. 28, no. 3, pp. 1723–1730, Apr. 2013.

[39] G. P. Adam *et al.*, “Modular multilevel inverter: Pulse width modulation and capacitor balancing technique,” *IET Power Electron.*, vol. 3, no. 5, pp. 702–715, Sep. 2010.

[40] S. Priyadarshi *et al.*, “Parallel transient simulation of multiphysics circuits using delay-based partitioning,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 31, no. 10, pp. 1522–1535, Oct. 2012.

[41] G. S. Murthy, M. Ravishankar, M. M. Baskaran, and P. Sadayappan, “Optimal loop unrolling for GPGPU programs,” in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, Apr. 2010, pp. 1–11.

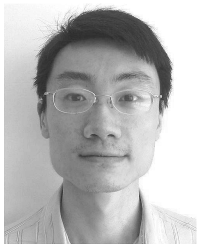
[42] J. S. Przemieniecki, “Matrix structural analysis of substructures,” *AIAA J.*, vol. 1, no. 1, pp. 138–147, 1963.

[43] H. A. Schwarz, *Gesammelte Mathematische Abhandlungen*, vol. 2. Berlin, Germany: Springer, 1890, pp. 133–143.

[44] E. Solomonik and J. Demmel, “Communication-optimal parallel 2.5 D matrix multiplication and LU factorization algorithms,” Dept. Elect. Eng., Comput. Sci., Univ. California, Berkeley, CA, USA, Tech. Rep. EECS-2011-10, Feb. 2011.

[45] L. Chen, O. Villa, S. Krishnamoorthy, and G. R. Gao, “Dynamic load balancing on single- and multi-GPU systems,” in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, Atlanta, GA, USA, Apr. 2010, pp. 1–12.

[46] J. L. Gustafson, “Reevaluating Amdahl’s law,” *Commun. ACM*, vol. 31, no. 5, pp. 532–533, 1988.



ZHIYIN ZHOU (S’10) received the B.Sc. degree in electronic science and engineering from Nanjing University, Nanjing, China, in 2000, and the M.Sc. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 2012, where he is currently working toward the Ph.D. degree. His research interests include large-scale parallel and distributed computing, massively parallel programming, power system simulation, and electromagnetic transient studies.

VENKATA DINAVAHI (S’95–M’00–SM’08) received the Ph.D. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2000. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. His research interests include real-time simulation of power systems and power electronic systems, largescale system simulation, and parallel and distributed computing.