Matrix-Free Nonlinear Finite-Element Solver Using Transmission-Line Modeling on GPU

Peng Liu, Jiacong Li, and Venkata Dinavahi

Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2R3, Canada

The transmission-line modeling (TLM) used for nonlinear finite-element (FE) solution has a paramount feature that the admittance matrix is unchanged and only needs one-time factorization; and this feature becomes a drawback when the required number of TLM iterations increase due to the mismatch between the transmission-line impedance and the load. In this paper, a matrix-free TLM scheme is proposed to make use of the solved nonlinear reluctivities without employing any matrices at each timestep, thus substantially decreasing the number of required TLM iterations. The matrix-free solver is suitable for massively parallel processing and the design is implemented on the Tesla V100 graphics processing unit (GPU). A speedup of more than 27 times is obtained compared with a commercial FE package for different problem sizes while maintaining high accuracy.

***Index Terms*— Distributed algorithms, domain decomposition, electromagnetic apparatus, finite-element (FE) method, graphics processing units (GPUs), massively parallel, nonlinear system of equations, transmission-line modeling (TLM).**

I. Introduction

THE magnetodynamics in electromagnetic apparatus with nonlinear B–H curve can be solved with the finite-element (FE) method [1], which is accurate but computationally expensive. In the past decades, the advent of graphics processing units (GPUs), such as the Tesla V100 accelerating card with 5120 cuda cores, has motivated researchers to explore massively parallel algorithms in order to make full use of the computing power.

The commonly used Newton–Raphson (NR) iterative scheme has to repeatedly solve a sparse linear system due to the updated Jacobian matrix, requiring efficient sparse solvers. Recently, the conjugate-gradient-based solvers have been attempted and implemented on GPU to improve the computational efficiency [2]–[4]. Restricted by the sequential portion of the program, the goal of massive parallelism can be stymied. The emerging trend of matrix-free FE method, which has massive parallelism and perfect computational load balance, has been employed for both linear [5], [6] and nonlinear FE problems [7] on GPU.

The transmission-line modeling (TLM), based on Huygens's principle for wave propagation, was proposed to solve nonlinear lumped networks [8] and later extended to nonlinear FE problems [9]–[11]. The TLM can decouple the nonlinear elements from the network so that the nonlinearities can be solved iteratively in a massively parallel manner, and the unchanged admittance matrix factorized at the beginning is a paramount feature to reduce the overall computational costs. Due to the mismatched impedances of the transmission-line and nonlinear load, usually hundreds or even thousands of TLM iterations are required; using an adaptive admittance matrices will decrease the required TLM iteration number, whereas it will naturally incur additional computational cost for factorizing matrices, which has been a bottleneck of the TLM applied in nonlinear FE problems.

In this paper, a nonlinear 2-D magnetodynamic problem is solved by the FE-TLM scheme on Tesla V100 GPU with massive parallelism without employing any matrices. The adaptive transmission-line impedances are updated according to the solution of the previous timestep, so that the mismatch between the transmission-line impedance and the load impedance is greatly reduced, resulting in only a few (usually less than 10) TLM iterations. Instead of reassembling the transmission-line impedances to an admittance matrix and then factorizing it at each timestep, the linear network is solved using a matrix-free scheme, which is mathematically equivalent to the Jacobi iterative scheme and is insensitive to the change of transmission-line impedances. Massive parallelism is achieved in each phase of the matrix-free FE-TLM scheme to make full use of GPU's compute power, and the computational efficiency and accuracy are evaluated with regard to the commercial FE package Comsol MultiPhysics.

This paper is organized as follows. Section II introduces the details of the FE-TLM scheme, and Section III describes the adaptive transmission-line impedances and the matrix-free solver. Section IV provides the case study, the detailed implementation on GPU, and the result comparison. Section V gives the conclusion.

II. Finite-Element Transformer Model

A. Galerkin Finite-Element Method

A 2-D nonlinear magnetodynamic problem can be defined by the following diffusion equation:

$$\nabla \cdot (\upsilon \nabla A) = \sigma \frac{\partial A}{\partial t} - J_z \tag{1}$$

where A is the z-component of the magnetic vector potential to be solved, υ is the field-dependent reluctivity, σ is the conductivity, and J is the z-component of the impressed current density.

Manuscript received November 6, 2018; revised February 25, 2019; accepted March 1, 2019. Date of publication March 21, 2019; date of current version June 20, 2019. Corresponding author: P. Liu (e-mail: pliu3@ualberta.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2019.2903008

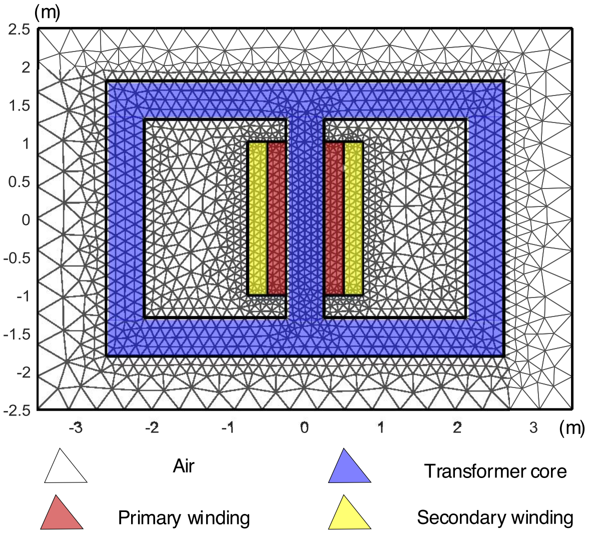

Fig. 1. Problem domain and FE mesh of a 2-D transformer.

The problem domain of an E-core transformer model is shown in Fig. 1. According to the Galerkin method, the integral of the weighted residual over each element $\boldsymbol{\Omega}^e$ can be written as

$$\iint_{\Omega^e} \upsilon^e \left(\frac{\partial A^e}{\partial x}\frac{\partial W^e}{\partial x} + \frac{\partial A^e}{\partial y}\frac{\partial W^e}{\partial y}\right) dxdy + \iint_{\Omega^e} \sigma \frac{\partial A^e}{\partial t} W^e dxdy = \iint_{\Omega^e} J_z^e W^e dxdy \quad (2)$$

where A^e represents the local linear interpolation within each triangular element. Setting the weighted function W^e to the shape functions, then the elemental equations are obtained

$$\frac{\upsilon^e}{4\Delta^e}\begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{31} & k_{33} \end{bmatrix}\begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} + \frac{\sigma^e\Delta^e}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}\begin{bmatrix} \frac{\partial A_1}{\partial t} \\ \frac{\partial A_2}{\partial t} \\ \frac{\partial A_3}{\partial t} \end{bmatrix} = \frac{J_z^e\Delta^e}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}. \quad (3)$$

Due to the nonlinear reluctivity, these elemental equations are generally assembled to a global nonlinear system of equations and solved by the NR iterative scheme.

The TLM solution of the nonlinear FE problem is quite different.

B. TLM Solution of Finite-Element Model

In fact, (3) defines an equivalent network composed of capacitors and nonlinear resistors shown in Fig. 2(a). The values of the components are given as

$$G_{12} = -\frac{\upsilon^e}{4\Delta^e}k_{12}, \quad G_{13} = -\frac{\upsilon^e}{4\Delta^e}k_{13}, \quad G_{23} = -\frac{\upsilon^e}{4\Delta^e}k_{23}$$
$$Y_{G12} = -\frac{\upsilon_g^e}{4\Delta^e}k_{12}, \quad Y_{G13} = -\frac{\upsilon_g^e}{4\Delta^e}k_{13}, \quad Y_{G23} = -\frac{\upsilon_g^e}{4\Delta^e}k_{23}$$
$$C_{12} = C_{13} = C_{23} = -\frac{\sigma^e\Delta^e}{12}, \quad Y_{C12} = Y_{C13} = Y_{C23} = -\frac{\sigma^e\Delta^e}{6\Delta t}$$

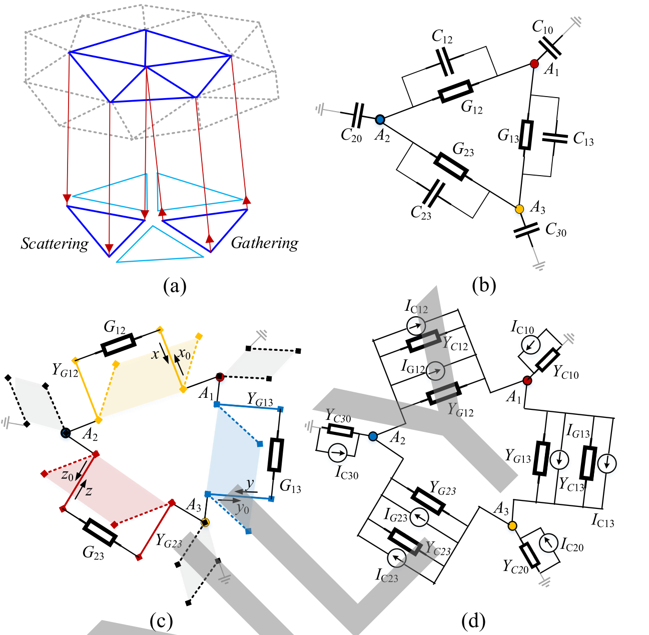

Fig. 2. TLM technique applied to elemental equations of the FE method [10]. (a) 2-D triangular mesh. (b) Equivalent nonlinear network. (c) TLM model: scattering. (d) TLM model: gathering.

$$C_{10} = C_{20} = C_{30} = \frac{4\sigma^e\Delta^e}{12}, \quad Y_{C10} = Y_{C20} = Y_{C30} = \frac{4\sigma^e\Delta^e}{6\Delta t}. \quad (4)$$

As shown in Fig. 2, imagine the virtual transmission lines separate these resistors and capacitors, thus each TLM iteration will include two phases.

In the scattering phase shown in Fig. 2(c), the incident pules enter each component from the network, and the reflected pulses can be obtained independently within each triangular element by solving a 3 × 3 nonlinear system of equations.

In the gathering phase shown in Fig. 2(d), the reflected pulses return to the network and react with each other. In order to obtain the next incident pulses, a linear network needs to be solved by replacing the TLM links or stubs with their equivalent Norton circuits.

III. Adaptive Characteristic Impedances and Matrix-Free Solver

A. Adaptive Transmission-Line Impedances

Note that in (4), υ_g^e is a guessed value of the real υ^e that is unknown, and the mismatch will cause the scattering and gathering phases to repeat many iterations to converge. In practice, the guessed value of υ_g^e is usually set to the υ of the linear portion of the B–H curve. In addition, the admittance matrix of the linear network is determined by the characteristic impedance of the transmission line regardless of the nonlinear resistors, thus the admittance matrix usually remains unchanged.

The unchanged admittance matrix implies that only one lower-upper (LU) decomposition (or matrix inversion) at the beginning of the program is enough, and for the gathering phase, the remaining numerical operations to solve the linear

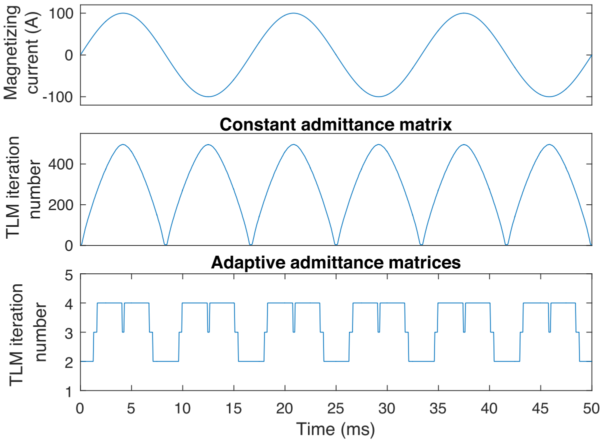

Fig. 3. Magnetizing current, required number of TLM iterations with constant admittance matrix, and required number of TLM iterations with adaptive admittance matrices.

network are mere backward and forward substitution (or matrix-vector multiplication). However, in real applications, the real value of the nonlinear reluctivity $\upsilon^e(t)$ is always time-varying. Thus, the extent of mismatch between the constant υ_g^e and the changing $\upsilon^e(t)$ will fluctuate, and the required number of TLM iterations may also vary from tens to thousands [11]. Therefore, although the classical TLM technique has perfect parallelism that can benefit from parallel processing in the scattering phase, the thousands of TLM iterations required and the sequential nature of the backward and forward substitution are the bottleneck to further improve the computational efficiency.

In fact, at any time point t, the solution includes the real values of $\upsilon^e(t)$ for all triangular elements. If this information is used to decrease the mismatch of the transmission-line impedance and the load to be solved, implying $\upsilon_g^e(t+\Delta t) = \upsilon^e(t)$, the required TLM iteration number for the next timestep $t+\Delta t$ will be substantially decreased because the values of $\upsilon_g^e(t+\Delta t)$ and $\upsilon^e(t+\Delta t)$ within each triangular element are very close. Fig. 3 shows that by setting the guessed transmission-line impedance to the solved impedance value from the previous timestep within each element, the number of TLM iterations required substantially decreased from several hundreds to only 2–4 for each timestep.

It is well-known that assembling and factorizing the admittance matrix is required to solve the linear network in the gathering phase. Therefore, adaptive transmission-line impedances will cause repeatedly assembling and factorizing the admittance matrices, which will cause paramount computational burden and is what the TLM methodology has been trying to avoid. Naturally, such a question will arise: is it possible to utilize adaptive transmission-line impedances and meanwhile circumvent burdensome matrix operation?

B. Matrix-Free Scheme

Recently, the emerging matrix-free scheme provides a perfect solution to the above-mentioned situation. It originated from the fact that although all the FE nodes are related as a whole system, the sparsity nature determines that each node is only directly related to its neighbors.

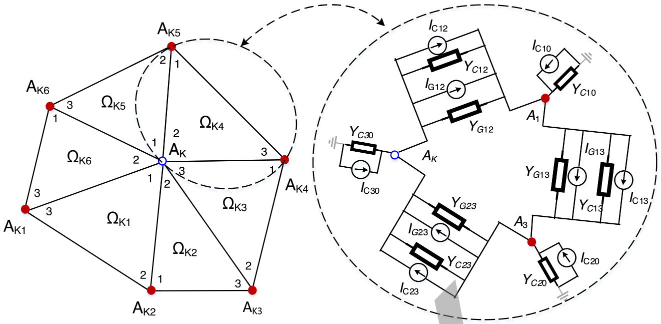

Fig. 4. Matrix-free solution scheme for one node.

Fig. 4 shows a sample FE node A_K and its surrounding nodes, and each triangular element is replaced with the equivalent Norton circuit in the scattering phase. Since the admittance matrix is sparse, it can be inferred that only a few entries are nonzero in the row where node A_K is the diagonal. The nonzero entries are corresponding with the neighboring nodes of A_K, ignoring the zero entries and the linear equation can be written as

$$k_{K1}A_{K1} + k_{K2}A_{K2} + k_{K3}A_{K3} + k_{K4}A_{K4} + k_{K5}A_{K5} + k_{K6}A_{K6} + \mathbf{k_K A_K} = b_K \quad (5)$$

where A_{Ki} represents the unknown nodal value, k_{Ki} is the stiffness coefficient, and b_K is the right-hand known quantity associated with the injected current source. The coefficients can be obtained by introducing the node-element connection information, for example, k_{K1} is composed of Y_{G13} in element Ω_{K1} and Y_{G23} in element Ω_{K6}, whereas $\mathbf{k_K}$ is composed of Y_{G11}, Y_{G22}, Y_{G33}, Y_{G22}, Y_{G11}, and Y_{G22} in element Ω_{K1}, Ω_{K2}, Ω_{K3}, Ω_{K4}, Ω_{K5}, and Ω_{K6}, respectively.

According to the diagonally dominant property of a FE stiffness matrix [1], the above-mentioned equation can be solved using the Jacobi relaxation scheme

$$\mathbf{A_K^{i+1}} = \frac{1}{\mathbf{k_K}}(b_K - k_{K1}A_{K1}^i - k_{K2}A_{K2}^i - k_{K3}A_{K3}^i - k_{K4}A_{K4}^i - k_{K5}A_{K5}^i - k_{K6}A_{K6}^i) \quad (6)$$

where i represents the iteration number.

Note that the above-mentioned process applies to every nonboundary (unknown) node and each node can be executed independently in parallel, meanwhile, a synchronization is required at the end of each iteration. The convergence condition used in this paper is that the maximum relative tolerance between two successive iterations is less than 10^{-5}. No matrix operations are involved at all, instead, it is essential to obtain the node-element connection information such as the indexes of a node's neighboring nodes, the indexes of all triangular elements surrounding each node, and how the nodes and elements are connected.

There are two major advantages by applying the matrix-free scheme to the gathering phase of the TLM technique. First, when applying the adaptive transmission-line impedances, i.e., when Y_{Gij} (ij = 12, 13, 23) in (4) change, matrix assembling or factorizing are avoided, instead, the coefficients in (5) need to be updated within each element in a massively

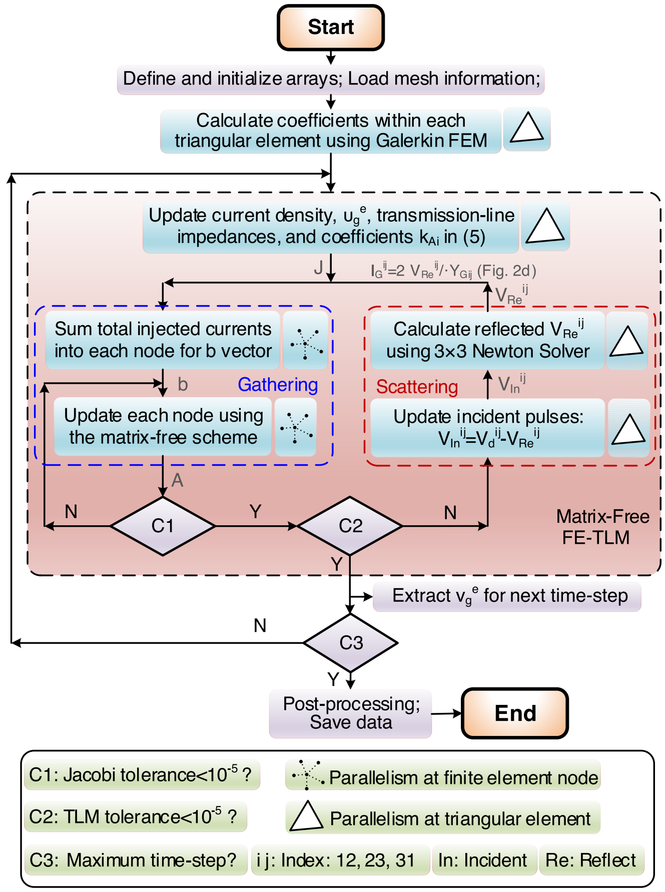
Start
Define and initialize arrays; Load mesh information;
Calculate coefficients within each triangular element using Galerkin FEM
Update current density, u_g^e, transmission-line impedances, and coefficients k_{Ai} in (5)
J
$I_G^{ij}=2\ V_{Re}^{ij}/\cdot Y_{Gij}$ (Fig. 2d)
V_{Re}^{ij}
Sum total injected currents into each node for b vector
Calculate reflected V_{Re}^{ij} using 3×3 Newton Solver
b
Gathering
Scattering
V_{In}^{ij}
Update each node using the matrix-free scheme
Update incident pulses: $V_{In}^{ij}=V_d^{ij}-V_{Re}^{ij}$
A
N
C1
Y
C2
N
Matrix-Free FE-TLM
Y
Extract v_g^e for next time-step
N
C3
Y
Post-processing; Save data
End
C1: Jacobi tolerance<10^{-5} ?
Parallelism at finite element node
C2: TLM tolerance<10^{-5} ?
Parallelism at triangular element
C3: Maximum time-step?
i j: Index: 12, 23, 31
In: Incident
Re: Reflect

Fig. 5. Detailed implementation of the matrix-free TLM scheme on GPU.

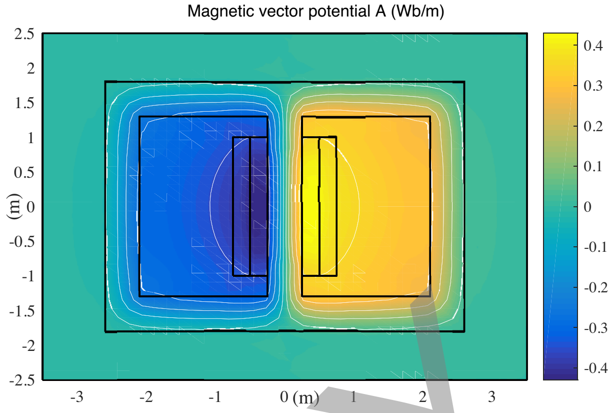

Fig. 6. Distribution of magnetic vector potentials from the matrix-free TLM, when the magnetizing current is peak value (100 A).

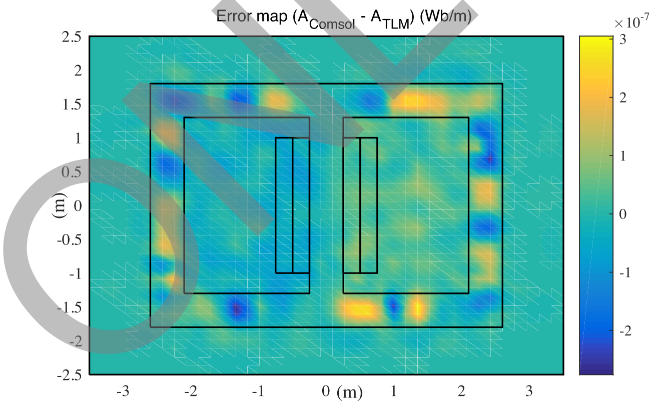

Fig. 7. Error map of the magnetic vector potentials.

parallel manner and the remaining iterative scheme is not sensitive to the change of impedances. In addition, the repeated sequential backward and forward substitution, which can be hardly parallelized, is also avoided and the matrix-free scheme has massive parallelism. Therefore, by applying the matrix-free scheme, the required TLM iterations are substantially reduced and at the same time, both the scattering phase and the gathering phase can be executed on massively parallel computing resources such as GPU with perfect computational load balance because all nodes or elements are treated equally. Thus, the overall computational efficiency can be considerably improved by sufficiently exploring the massively parallel computing resources.

IV. Case Study and Results

A. Case Setup and Parallel Implementation

The 2-D transformer model given in Section II is selected for the case study. The magnetizing current is shown in Fig. 3, and the nonlinear B–H curve can be found in [12]. The relative tolerance of the matrix-free solver and the TLM tolerance are set to 10^{-5} to ensure convergence, and the applied timestep is $50\mu s$.

The detailed program flow of the implementation on GPU is illustrated in Fig. 5. At the beginning of each timestep, the transmission-line impedances and the coefficients k_{Ai} in (5) associated with v_g^e are updated based on the solution of the previous timestep. The scattering phase, gathering phase, conditions that control the program flow, and the parallelism at the nodal or elemental level for each block are all illustrated.

B. Accuracy and Speedup Evaluation

The computational efficiency and accuracy of the proposed matrix-free FE-TLM scheme are evaluated with the simulation results from commercial software Comsol as a benchmark. The Comsol simulation was carried out on a workstation with dual Intel Xeon E5-2698 v4 CPUs, 20 cores each, 2.2 GHz clock frequency, and 128 GB RAM. The available number of cores for the simulation was set to 40. The matrix-free FEM-TL codes were executed on the NVIDIA Tesla V100 GPU with 5120 Cuda cores.

Fig. 6 shows the distribution of the magnetic vector potentials obtained from the matrix-free TLM scheme on GPU and Fig. 7 shows the error map compared with those results from Comsol for a case study with 509 nodes and 967 elements, proving the high accuracy of the TLM scheme. It can also be inferred that other parameters such as magnetic flux density $(B = \nabla \times A)$ and magnetic field strength H can be also obtained after post-processing with an error of the same level.

TABLE I

MATRIX-FREE TLM EXECUTION TIME AND SPEEDUP ON GPU PARALLELIZATION

Cases	Number of Nodes	Comsol^{TM} Execution Time (s)	Non-adaptive TLM with Many Iterations Execution Time (s)	Adaptive TLM Involving Matrices Execution Time (s)	Matrix-Free TLM on GPU	
					Execution Time (s)	Speedup
Case 1	509	0.72	0.21	0.345	0.026	27.74
Case 2	1273	1.92	0.47	1.04	0.068	28.2
Case 3	3303	6.85	1.24	3.95	0.214	32.0
Case 4	4923	9.30	1.84	4.60	0.335	27.8
Case 5	9784	14.1	2.96	9.15	0.633	22.3

Table I shows the runtime of the Comsol nonlinear solver, the classical TLM scheme without adaptive impedances (many TLM iterations required), the TLM scheme with adaptive impedances involving matrices (LU decomposition required at each timestep), and the matrix-free TLM implemented on GPU implementation. It can be concluded that by exploring parallelism in the scattering phase, traditional TLM scheme with constant admittance matrix can achieve a speedup of $3-5$ times compared with Comsol. After applying the adaptive impedances, although the required TLM iterations are greatly reduced, the matrix assembling and decomposing required at each timestep become the main computational burden, causing even more execution time. However, with the matrix-free scheme implemented on Tesla V100 GPU, a speedup of more than 27 times is achieved for five different problem sizes by fully exploring the parallelism and, meanwhile, avoiding the sequential and burdensome matrix operations. The runtime almost has a linear increase with the problem size.

It should be mentioned that in Table I, the speedup slightly drops for the Case 5 because the number of nodes (9784) exceeds the number of Cuda cores (5120), implying that most Cuda cores have to compute two nodes in sequential. Therefore, the massively parallel computer resource that matches the FE problem size is a prerequisite to fully explore the computational power of the proposed matrix-free FE-TLM scheme.

V. CONCLUSION

A matrix-free TLM scheme is proposed and implemented on Tesla V100 GPU with massive parallelism for a nonlinear FE transformer model. For each timestep, the nonlinear reluctivities are extracted as transmission-line impedances to substantially decrease the required number of TLM iterations. The linear network is solved using a matrix-free scheme to make use of the transmission-line impedances without having to assemble and factorize the admittance matrix. All phases of the TLM procedure are perfectly parallelized with good load balance, and the runtime on GPU is more than 27 times faster than Comsol for different problem sizes.

The proposed matrix-free TLM scheme can also be extended to 3-D FE model using nodal or edge elements.

VI. ACKNOWLEDGMENT

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

REFERENCES

[1] J.-M. Jin, *The Finite Element Method in Electromagnetics*, 3rd ed. Hoboken, NJ, USA: Wiley, 2015.

[2] A. F. P. de Camargos, V. C. Silva, J-M. Guichon, and G. Munier, "Efficient parallel preconditioned conjugate gradient solver on GPU for FE modeling of electromagnetic fields in highly dissipative media," *IEEE Trans. Magn.*, vol. 50, no. 2, Feb. 2014. Art. no. 7014004.

[3] T. Okimura, T. Sasayama, N. Takahashi, and S. Ikuno, "Parallelization of finite element analysis of nonlinear magnetic fields using GPU," *IEEE Trans. Magn.*, vol. 49, no. 5, pp. 1557–1560, May 2013.

[4] M. M. Dehnavi, D. M. Fernandez, and D. Giannacopoulos, "Enhancing the performance of conjugate gradient solvers on graphic processing units," *IEEE Trans. Magn.*, vol. 47, no. 5, pp. 1162–1165, May 2011.

[5] J. P. A. Bastos and N. Sadowski, *Magnetic Materials and 3D Finite Element Modeling*. Nashville, TN, USA: CRC Press, 2013.

[6] D. M. Fernandez, M. M. Dehnavi, W. J. Gross, and D. Giannacopoulos, "Alternate parallel processing approach for FEM," *IEEE Trans. Magn.*, vol. 48, no. 2, pp. 399–402, Feb. 2012.

[7] P. Liu and V. Dinavahi, "Matrix-free nodal domain decomposition with relaxation for massively parallel finite-element computation of EM apparatus," *IEEE Trans. Magn.*, vol. 54, no. 9, Sep. 2018. Art. no. 7402507.

[8] P. B. Johns and M. O'Brien, "Use of the transmission-line modelling (T.L.M.) method to solve non-linear lumped networks," *Radio Electron. Eng.*, vol. 50, nos. 1.2, pp. 59–70, Jan. 1980.

[9] O. Deblecker, J. Lobry, and C. Broche, "Use of transmission-line modelling method in FEM for solution of nonlinear eddy-current problems," *IEE Proc.-Sci., Meas. Technol.*, vol. 145, no. 1, pp. 31–38, Jan. 1998.

[10] P. Liu and V. Dinavahi, "Real-time finite-element simulation of electromagnetic transients of transformer on FPGA," *IEEE Trans. Power Del.*, vol. 33, no. 4, pp. 1991–2001, Aug. 2018.

[11] O. Deblecker, J. Lobry, and C. Broche, "Novel algorithm based on transmission-line modeling in the finite-element method for nonlinear quasi-static field analysis," *IEEE Trans. Magn.*, vol. 39, no. 1, pp. 529–538, Jan. 2003.

[12] [Online]. Available: http://magweb.us/free-bh-curves/.

Peng Liu (S'15) was born in Xuchang, Henan, China, in 1992. He received the B.Sc. and M.Eng. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2013 and 2015, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada.

His current research interests include computational electromagnetics, finite-element analysis, and parallel and distributed processing.

Jiacong Li (S'18) was born in Ningxia, Yinchuan, 1994. He received the B.Sc. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada.

His current research interests include high-frequency computational electromagnetics and parallel finite-element method.

Venkata Dinavahi (S'94–M'00–SM'08) received the Ph.D. degree 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 current research interests include real-time simulation of power systems, large-scale system simulation, and parallel and distributed computing.