Parallel 3-D Finite Element Modeling for Electromagnetic Transient Simulation

by

Jiacong Li

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in
Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Electromagnetic devices, such as transformers and power reactors, are widely used in power systems. Traditionally, these essential components are simplified into lumped networks in power system electromagnetic transient (EMT) simulations. However, the traditional lumped network model cannot always accurately represent the transient behaviors of these inductive components, especially when complex physical phenomena are encountered (such as magnetic saturation, eddy currents, and hysteresis). On the other hand, the finite element method (FEM) has become a powerful tool to solve complex physical fields, due to its superior accuracy, and the ability to handle complex geometries and material properties. Therefore, researchers have been giving increasing attention to finite element models for energy system EMT simulations in recent years to achieve high precision designs.

Despite the excellent accuracy, finite element models, comparing with lumped networks, require a dramatically larger amount of computational power. This is a result of the following facts. The device space and electromagnetic field are discretized into many smaller elements and innumerable degree of freedoms (DOF or unknowns) to be solved. This leads to an 'oversized' nonlinear matrix system. Also, to handle the nonlinearity, traditional Newton-Rapson (NR) solvers need to repetitively form and factorize the large matrix system, which significantly slows down the simulation.

With the fast development of parallel computing hardware, researchers have recently introduced the transmission line modeling (TLM) and the nodal domain decomposition method to boost 2-D finite element models. These novel ideas eliminate the traditional repetitive matrix assembly (forming) and factorization during the NR solving process, and their DOF-level parallelism leads to significant speed-up compared with commercial FEM software. However, the above fast algorithms are based on 2-D nodal finite elements, while in reality, power system devices are in 3-D geometry, and simplification from 3-D to 2-D causes excessive information loss. Also, the nodal element leads to significant errors around sharp geometries. 3-D edge finite elements, in contrast, do not have the above

shortages, while similar algorithms are rarely seen in 3-D edge FEM.

In this thesis, the above high-performance massively-parallel methods are further extended to 3-D edge-finite-elements-based EMT simulation models. Challenges due to the different nature between 2-D nodal and 3-D edge element formulations are successfully tackled, and parallelism is explored from different perspectives.

First, the TLM algorithm is developed to solve nonlinear 3-D edge element problems with parallelism explored at each elements' viewpoint. Transmission lines are deployed to decouple each element's local nonlinearity from the global 'oversized' matrix. Benefitting from such isolation, the method only requires massively-parallelized small local Newton-Rapson iterations in each element, which is perfectly suitable for GPU architectures. The decoupling also allows constant global admittance matrix. Thus, the repetitive formation and factorization process of the large global matrix are avoided.

Second, from edge's view, the edge domain decomposition (EDD) method is proposed to parallelize nonlinear 3-D edge element problems at edge DOF level. The electromagnetic device's entire space is divided into many sub-domains that only contain one edge unknown. The solution of light-weight nonlinear sub-domain systems can be massively parallelized, and the neighbor-to-neighbor communication scheme eliminates the need to form a global finite element matrix. From the perspective of time, the well-known parallel-in-time method is also adapted to the EDD FEM system to achieve the space-time parallelism.

In addition, the thesis also proposes a novel coupling scheme to interface the 3-D finite element models with external circuits under large eddy currents without having to combine circuit and FEM system in one matrix. Implementations of these massively-parallel algorithms indicate excellent efficiency and accuracy.

Preface

The materials presented in the thesis are original intellectual work created by Jiacong Li. All the algorithms and implementations are developed and tested in the Real-Time Experimental Laboratory (RTX-Lab) at the University of Alberta. Jiacong Li carried out the majority of the work including model and method derivation, program development and test, and thesis composition under the supervision of Dr. Venkata Dinavahi. Peng Liu also contributed to the research and thesis by providing suggestions and discussions about the idea of the models, thesis content, and figures of the 2-D FEM background. Some parts of the thesis have been published in IEEE journal.

Chapter 3 includes contents from the following paper:

- J. Li, P. Liu and V. Dinavahi, "Massively Parallel Computation for 3-D Nonlinear Finite Edge Element Problem With Transmission Line Decoupling Technique", *IEEE Transactions on Magnetics*, vol. 55, no. 10, pp. 1-8, Oct. 2019.

Chapter 4 includes contents from the following paper:

- J. Li, P. Liu and V. Dinavahi, "Matrix-Free Edge Domain Decomposition Method for Massively Parallel 3-D Finite Element Simulation with Field-Circuit Coupling", *IEEE Transactions on Magnetics*, pp. 1-9, July 2020, doi: 10.1109/TMAG.2020.3013143.(early access)

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Venkata Dinavahi for providing kind support and innovative research ideas during my study at the University of Alberta. His professional instructions and passionate altitudes motivate me to overcome many challenges in conducting this research topic.

I also want to express thanks to my friends in the RTX-lab including Peng Liu, Tian Liang, Qingjie Xu, and Tianshi Cheng. They have provided valuable suggestions for both my academic and personal life.

Love to my parents and family!

Table of Contents

List of Tables

List of Figures

List of Acronyms

CPU	central processing unit
CUDA	compute unified device architecture
DOF	degrees of freedom
EDD	edge domain decomposition
EMT	electromagnetic transient
EVE	edge vector elements
FE	finite element
FEM	finite element method
MEC	magnetic equivalent circuit
GMRES	generalized minimal residual
GPU	graphics processing units
MVP	magnetic vector potential
NR	Newton Rapson
NDD	Nodal Domain Decomposition
NDDR	nodal domain decomposition relaxation
NSE	nodal scalar elements
ODE	ordinary differential equations
PARDISO	PARallel DIrect sparse SOlver
RMVP	reduced magnetic vector potential
SIMD	single instruction multiple data
SM	steaming modules
TLM	transmission line modeling

1 Introduction

The electromagnetic transient (EMT) simulation has been an essential tool for the design, test, and engineering of power and energy system. For higher precision of power system EMT simulation, finite element models have witnessed increasing attention during the past years to replace traditional lumped power system devices. Benefiting from the development of high-performance computation, fast and accurate 3-D finite element models for power system simulations become possible.

This chapter introduces the background of carrying out the research topic of developing massively-parallel algorithms for 3-D finite element device models in power system simulations. The importance of the finite element method for high precision simulation is explained first. The reason for the high computation burden for traditional finite element models is also discussed. The chapter then introduces the development of parallel computing hardware and some massively-parallelized techniques to accelerate 2-D finite element simulations. The motivation and challenge of expending these methods to 3-D finite element models and the outline of the thesis are explained at last.

1.1 Importance of Finite Element Method in EMT Simulation

Electromagnetic devices, such as transformers and power reactors, are essential components in power and energy systems. The electromagnetic field inside these devices establishes the connections between their terminal voltages, currents, and the geometry of the transformer materials. Such relations are subject to Maxwell's equations, and how to accurately and efficiently model this relation is an important topic in electromagnetic transient (EMT) simulations of power and energy systems.

Traditionally, the voltage-current coupling relation for these essential power system

devices is simplified into lumped models [1–6], which have been widely applied in power system EMT simulation programs, and these network models can be classified into admittance matrix representation, equivalent-electrical-circuit representation, and the magnetic equivalent circuit (MEC) representation [7], [8]. However, for some electromagnetic devices such as transformers, motors, and generators, the traditional lumped network model cannot always accurately represent their transient behaviors, since the complex phenomena (such as material nonlinearity, eddy currents, and magnetic saturation) are always encountered in these devices. This is mainly caused by the following reasons. The equivalent-circuit/admittance matrix models omit the conductivity terms in Maxwell's equations so that the coil flux becomes purely linear to the coil current. For the MEC models, magnetic equivalent resistors cannot precisely represent the complex geometry and of the flux flow. Also, flux leakage cannot be accurately modeled. Moreover, all of these lumped parameters are usually assumed constant, which is not true for nonlinear materials with time-varying magnetic permeability. Therefore, some key information is lost during the field-to-lumped simplification process, these network models shy away for high-precision-requiring simulations.

On the contrary, finite element methods are known for their superior accuracy and ability to handle complex geometries and material properties, and the effectiveness of FEM has been verified in a variety of physical field problems such as structure, heat transfer, and electromagnetics analysis. In contrast with the above lumped models, the finite element method can directly solve Maxwell's equations for the electromagnetic field inside the electromagnetic devices, and the EM field distribution can be used to model the relation between voltage and current on device terminals. Thus, finite element models usually have the highest precision to represent electromagnetic devices, and researchers have been giving increasing attention to finite element models for energy system EMT simulations in recent years.

Despite the superior precision, finite element device models usually lead to an excessive computational burden. Such a huge computation cost is from the complex FEM solution process, which will be introduced in the next part (1.2).

1.2 General Process for FEM and Scope of This Thesis

For different physical boundary value problems, the finite element method shares a similar solution procedure. For a better explanation, a power inductor EMT problem is taken as an example. The electromagnetic field inside a power electromagnetic device is generated from currents inside the device coil. To find the electromagnetic field distribution, FEM follows the steps listed below [9]:

- Meshing: The space around the inductor is divided into many small elements (e.g. triangular or quadrangular for 2D, tetrahedral for 3D) as mesh, and the continuous

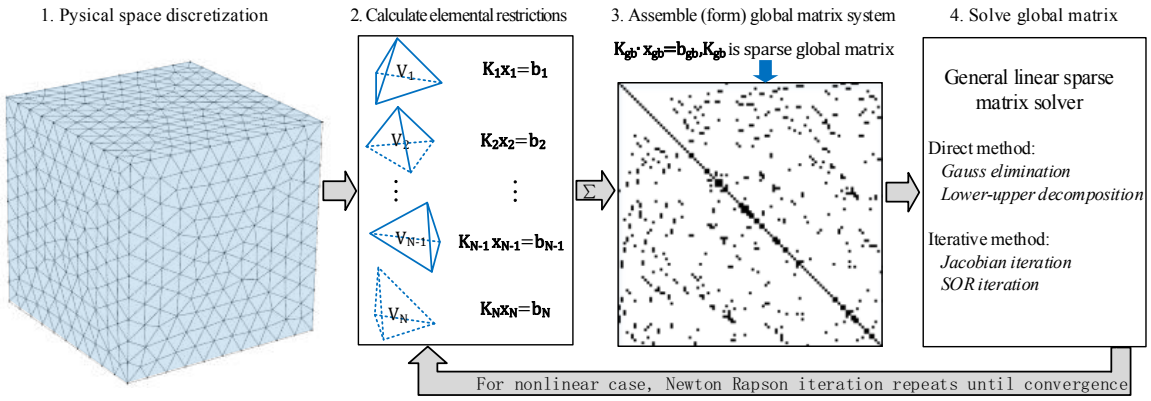


Figure 1.1: Procedure of traditional finite element method.

electromagnetic field is represented by a limited number of degrees of freedom (DOF, unknowns to be found) on associated with each vertex or edge.

- Generate elemental local restriction: inside each element, form the local restrictions between the local coil current density and unknown field DOF associated with the element.
- Assembly: add each element's local restrictions of the previous step into the global matrix system, based on inter-element electromagnetic field continuity principles. :
- Global matrix solution: Trim the global matrix with boundary conditions (i.e. set values for field intensity on the outer boundary of the inductor physical space). Solve the large-scale global matrix via sparse matrix solvers for the DOF values (x).

  Note: for nonlinear cases, matrix K is dependent on x, and Newton Rapson (NR) iterations are deployed to find the consistent solution of x and K. The iteration repeats steps 2 ˜4 in Fig. 1.1 until x convergences to a settled result.
- Post-processing: Restore the electromagnetic field distribution from the DOF values, plot, and display.

Power inductors usually have iron magnetic materials in the coil cores to increase their inductance. When magnetic saturation appears in the coil cores, permeability is no longer constant, and the above nonlinear Newton Rapson iterations become inevitable. During the NR solution process, all of the above steps (except for the meshing) have to be repetitively carried out. Even worse, the system usually has over thousands of DOFs, and even more elements, which significantly slows down the assembly and solution process of the global matrix. Therefore, the global Newton Rapson iteration consumes a large amount of computation power and becomes the main bottleneck of finite element models in power system simulations.

Efficiently improving the efficiency of the above NR solution process will shrink the design and test cycles in real applications, especially for large scale and multi-parameter optimizing designs, and this thesis will focus on exploring algorithms to boost or replace the time-consuming global nonlinear NR iterations, especially for 3-D FEM models. Also, it is worth mentioning that the meshing and post-processing will not be discussed in this thesis. These functions are achieved with the help of $Comsol^{TM}$ and $Matlab^{TM}$.

The development of high-performance computing hardware provides possibilities for researchers to overcome the bottleneck of finite element models, and this will be introduced in part 1.3 and 1.4.

1.3 High-Performance Computing Hardware

1.3.1 Development of Hardware

The development of computation hardware has reached a bottleneck on the increment of single-core clock frequency during the past decade. For better overall performance, more and more processing units are integrated into a single chip to execute programs in parallel. Parallel computing has witnessed rapid development, and Multi-core central processing units (CPU) and general-purpose graphics processing units (GPU) are widely applied for high-performance computing at different levels.

For example, the well-known Compute Canada has a supercomputer system of over 2000000 processing cores capable of providing a throughput of more than 2P floating points per second [10]. Also, for smaller-scale high-performance computers, the $NVIDIA^{®}$ company has recently released several flag-ship GPU solutions with thousands of CUDA cores [11], which is shown in Fig. 1.2.

1.3.2 Cutting-Edge GPU Architecture Used in This Thesis

In this thesis, we will utilize the state-of-art Tesla V100 PCIe GPU and the compatible programming API to accelerate power system finite element computation models. The Tesla V100 GPU consists of 80 multi-processor steaming modules (SM), and Fig. 1.3 illustrates the inner view of a streaming multi-processors. Each SM has its own dispatchers to simultaneously execute instructions on its arithmetic logic units (Cuda cores, including 64 FP32 units, 64 INT units, and 32 FP64 units). This means that the entire V100 GPU is able to carry out 5120 single-precision and 2560 double-precision operations in parallel. Also, the 128KB shared memory allows instant communication and repetitive data access for all threads within each SM. The V100 GPU also provides 16GB HBM2 high bandwidth memory and NVLink for communication for slower inter-SM and inter-GPU communications.

The Cuda cores (arithmetic logic units) do not have their own instruction dispatcher, which implies that the GPU is designed for single instruction multiple data (SIMD) applications, and $NVIDIA^{®}$ has invented the compute unified device architecture (CUDA)

	Tesla V100 PCIe	Tesla V100 SXM2	Tesla V100S PCIe
GPU Architecture	NVIDIA Volta		
NVIDIA Tensor Cores	640		
NVIDIA CUDA® Cores	5,120		
Double-Precision Performance	7 TFLOPS	7.8 TFLOPS	8.2 TFLOPS
Single-Precision Performance	14 TFLOPS	15.7 TFLOPS	16.4 TFLOPS
Tensor Performance	112 TFLOPS	125 TFLOPS	130 TFLOPS
GPU Memory	32GB /16GB HBM2		32GB HBM2
Memory Bandwidth	900GB/sec		1134 GB/sec
ECC	Yes		
Interconnect Bandwidth	32 GB/sec	300 GB/sec	32 GB/sec
System Interface	PCIe Gen3	NVIDIA NVLink	PCIe Gen3
Form Factor	PCIe Full Height/Length	SXM2	PCIe Full Height/Length
Max Power Comsumption	250 W	300 W	250 W
Thermal Solution	Passive		
Compute APIs	CUDA, DirectCompute, OpenCL™, OpenACC		

Figure 1.2: Data sheet of $NVIDIA^{®}$ Tesla V100 series GPU [11].

APIs in C language for researchers to develop SIMD programs. After the GPU despatcher launches the single instructions (defined as kernel C functions in CUDA API), the same routines (defined in the kernel function) will be executed on multiple different Cuda cores (or threads) in order to apply the same operations to multiple different datasets (local variables in multiple different threads). A typical SIMD example is summing two different vectors into one. Considering the above dispatching process, the SIMD program structure becomes crucially important in order to fully utilize the GPU architectures.

Thus, parallel computing becomes an inevitable trend.

1.4 Parallel Algorithms to Accelerate FEM Simulation

Despite the rapid development of high-performance computation device, the many-core and multi-core hardware do not sufficiently guarantee improvement on the solution speed of finite element models. To fully boost the FEM computation, algorithms need to be developed and adapted according to the different hardware architectures, and a suitable parallel strategy becomes particularly important. For example, the SIMD program structure is required on GPU for best performance, and it remains a challenge for researchers to create

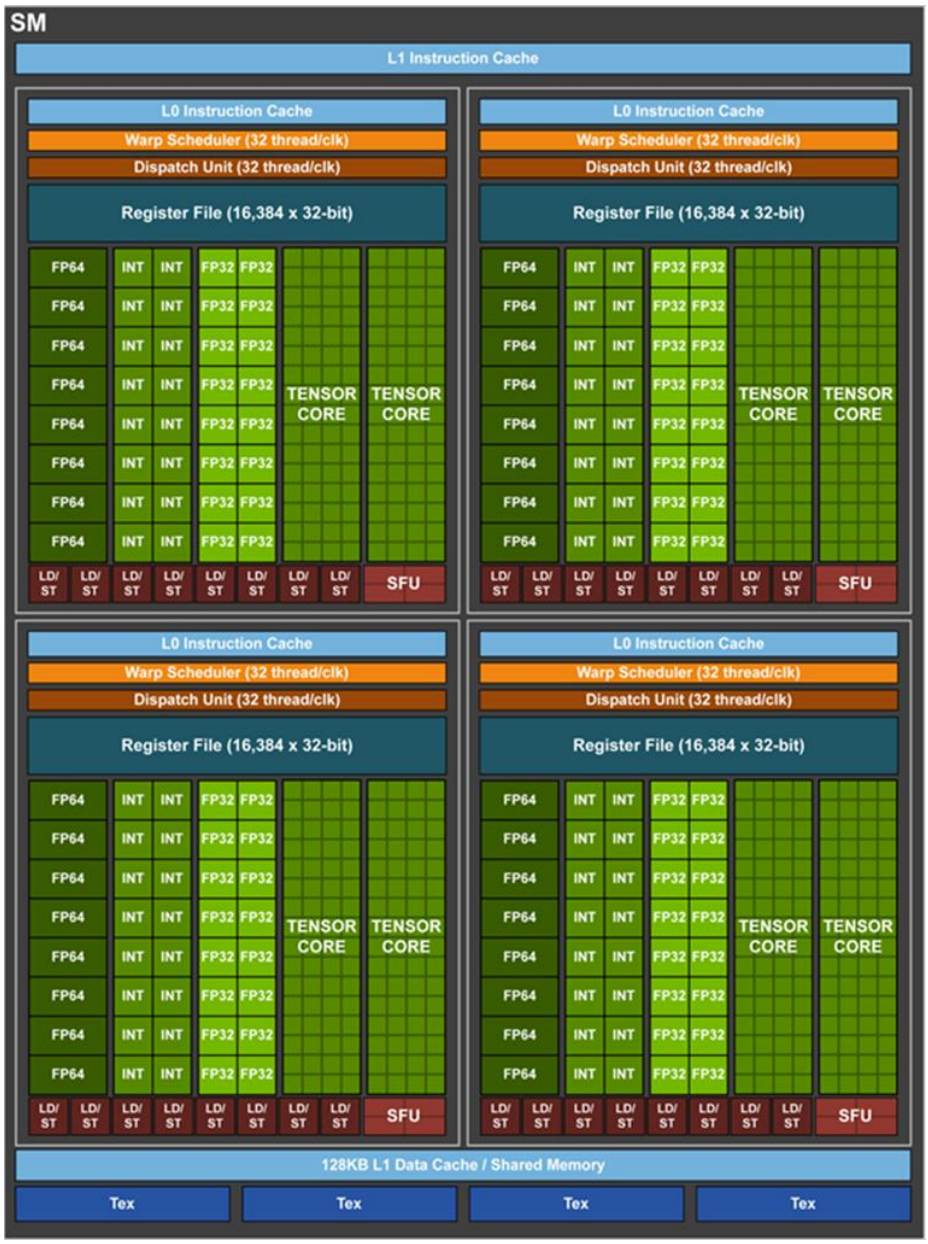


Figure 1.3: Architecture of streaming multi-processors in $NVIDIA^{®}$ Tesla V100 GPU [11].

finite element algorithms with suitable parallelism for different architectures, and many methods have been created in the past.

1.4.1 Traditional Parallelized Finite Element Computation

The domain decomposition method is one of the most commonly used technology to accelerate finite element computations. As is shown in Fig. 1.4, the whole solution region is divided into several sub-domains. Each sub-domain has its own matrix system, which is

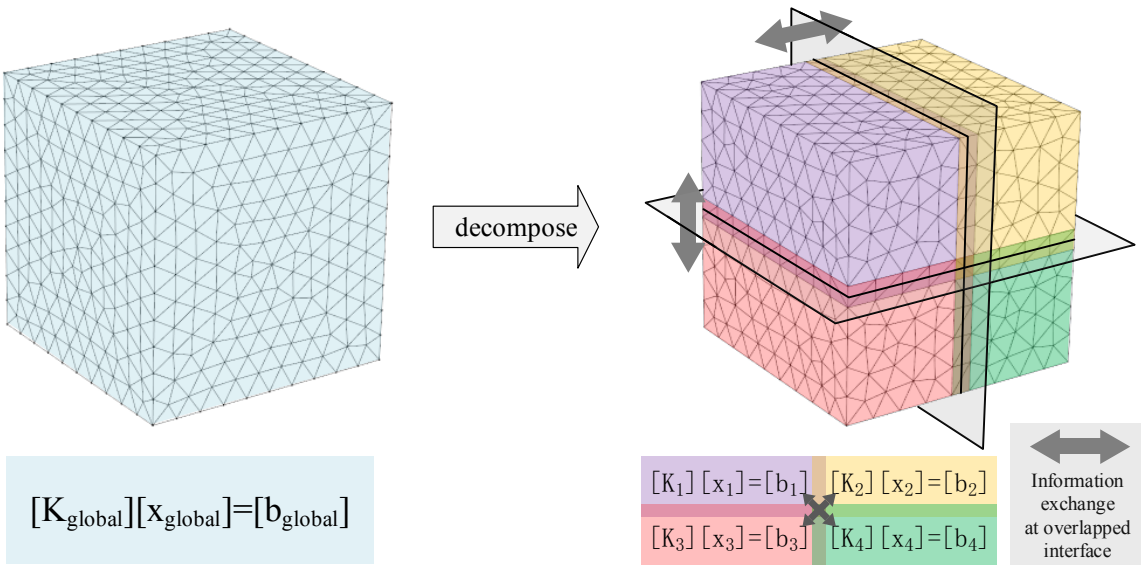


Figure 1.4: Domain decomposition scheme: global system and sub-domain systems.

much smaller than the global matrix system on the left. These sub-domain systems can be solved in parallel and different schemes [12], [13] may be applied to exchange information on sub-domain interfaces and reach the same solution as the global matrix system. Note that these information exchange schemes usually lead to much more computation amount vs. directly solving the global system, but since the sub-domain matrixes are solved in parallel, the overall execution time becomes smaller.

The global or sub-domain matrix systems can also be solved by parallelized sparse matrix solvers, such as the PARallel DIrect sparse SOlver (PARDISO) [14], and the generalized minimal residual (GMRES) iterative method [15], and significant speed-ups are achieved compared with their serial versions.

However, the above traditional parallel FEM methods are limited within the following stereotype: information is extracted from the distributed elements to generate a big-scale centralized matrix system. This makes sparse matrix solvers essential for the FEM computation. For many sparse solvers (such as PARDISO), branch structures are widely seen in different threads. Thus, it is hard to fully vectorize the computation process for SIMD-based GPU architectures. Also, the traditional FEM still cannot avoid the time-consuming Newton Rapson nonlinear iterations. During NR solving, the FEM matrix is not constant, which means local restriction generating, assembly, and sparse matrix solving repeats at each iteration step. As a result, a large amount of computation power is consumed.

In order to handle the above problems, researchers have recently introduced two innovative models for 2-D FEM [16], [17], and the NR iteration process is eliminated with a dramatic speed-up vs. $Comsol^{TM}$. This is introduced in 1.4.2 and 1.4.3.

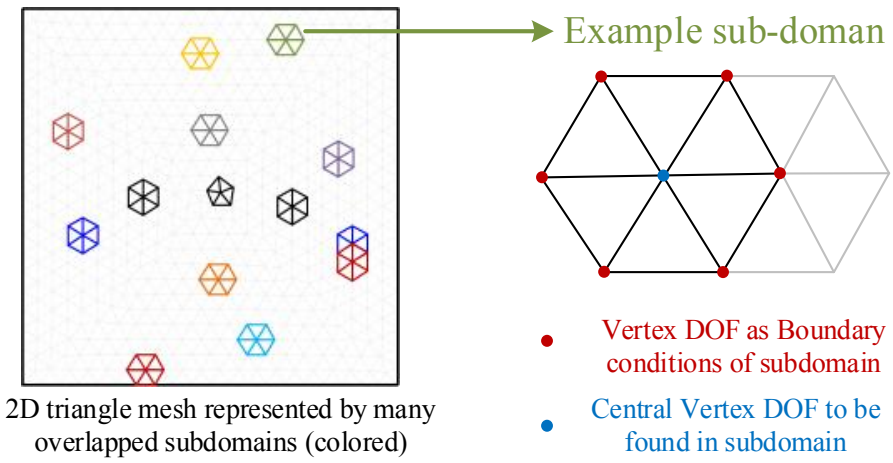


Figure 1.5: 2-D NDD scheme proposed in the previous studies.

1.4.2 Nodal Domain Decomposition (NDD) Method for 2-D Finite Element Models

Fig. 1.5 shows the nodal domain decomposition scheme. For a 2-D problem, the whole solution space is meshed into many triangular elements with each vertex associated with a degree of freedom to be found, and the entire mesh is represented as many small overlapped sub-domains composed of only one center vertex and all triangles around it. The sub-domain system is shown on the right, and each sub-domain forms a complete simple nonlinear boundary value problem: the red vertexes (nodes) are regarded as known boundary conditions of the BVP, and the blue node becomes the only unknown DOF.

An iteration scheme is applied to achieve a consistent solution of all DOF in the solution domain. At the start of the iterations, each node DOF is assigned an initial value, which serves as the red boundary condition in each sub-domain BVP. After that, all sub-domain are solved in parallel, generating the blue central DOF on each node. Due to overlapping these blue solutions then serves as the red boundary condition of their neighboring sub-domains for the next iteration step, and this iteration continues until all DOFs converges to the final solution.

During each iteration step, all 1-equation sub-domain BVP follows almost the same light-weight solution routine, with independency on each other. This makes the NDD scheme perfectly suitable for SIMD GPU architectures. Also, for the entire process, there is no matrix at all, thus the cumbersome global Newton Rapson iteration is successfully avoided.

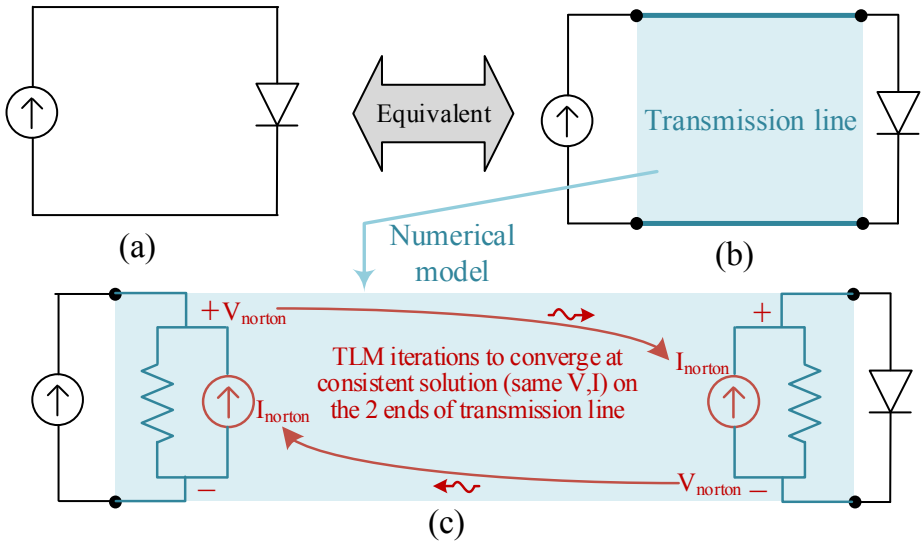


Figure 1.6: Procedure of traditional finite element method.

1.4.3 Transmission Line Modeling (TLM) Method for 2-D Finite Element Models

The transmission line was first introduced to isolate nonlinear resistors from linear networks in a circuit. The method origins form the fowling phenomenon: The insertion of an ideal transmission line between circuit components does not affect the steady-state voltage/current value on the two components, as is shown in Fig. 1.6(a) and (b). The voltage on the 2 ends of the transmission line may be different at the moment of insertion, but after traveling electromagnetic waves reflect and reach a steady state on the transmission line, the voltage becomes the same at the 2 ends. The process of reaching the steady state can be converted into a numerical model shown in Fig. 1.6(c). The insertion of the transmission line leads to isolation of the linear current source and the nonlinear diode, since both of them can only see traveling waves and constant characteristic impedance of the transmission line. The waves and characteristic impedance can be converted into 2 Norton sources with constant internal resistance at the 2 transmission line ends. Thus, the admittance matrix of the current source side becomes entirely constant, and a special iteration scheme can communicate information between the 2-ends Norton voltage and current, which will be introduced in the next chapters.

Based on the above discussion, inserting a transmission line at a nonlinear resistor can convert the nonlinear resistor into the equivalent model of a constant resistor in parallel with a current source, and this idea can be extended to finite element system. As is shown in Fig. 1.7, the finite element system is equivalent to a circuit which is formed by parallel-connected elemental local nonlinear resistors. If one inserts transmission lines on these resistors, they will become equivalent Norton sources, as is shown in Fig. 1.8. The constant resistors of the Norton sources in each element form a constant global resistor network,

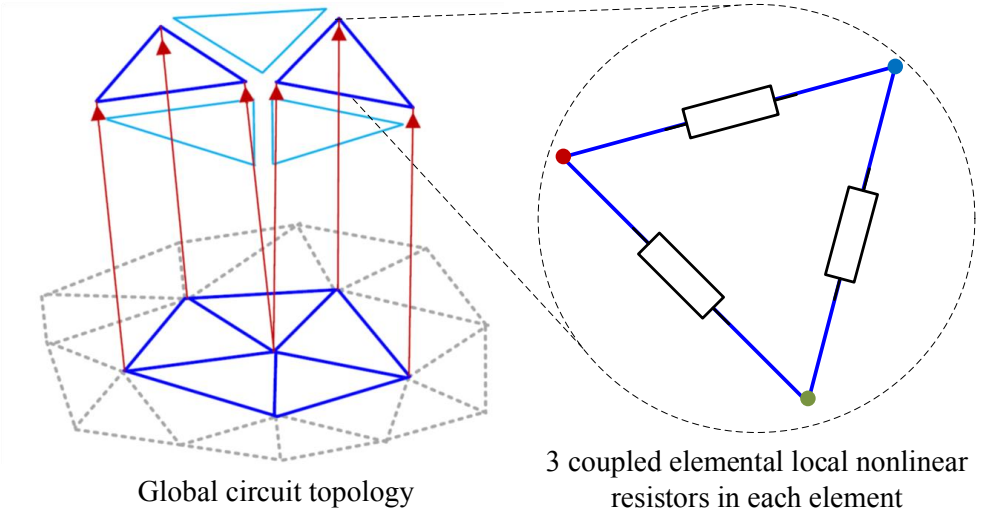


Figure 1.7: The equivalent circuit of 2-D FEM system.

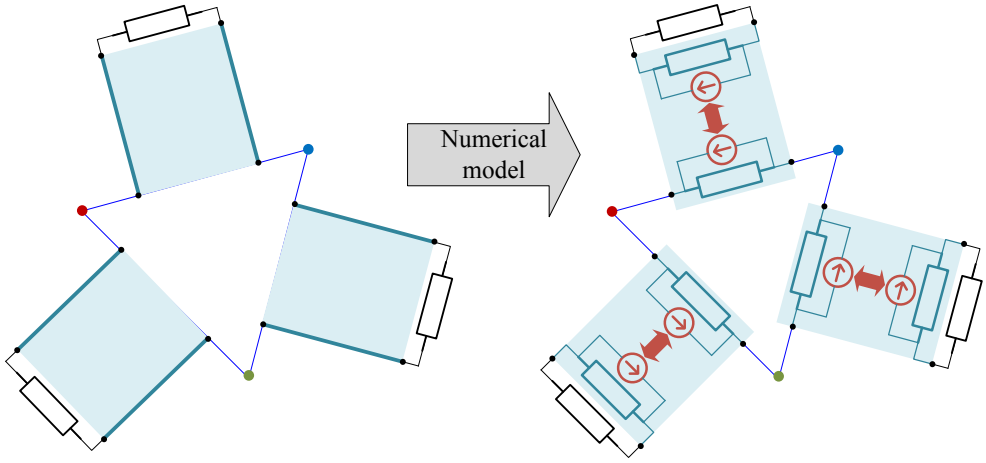


Figure 1.8: Elemental local nonlinear resistors decoupled by transmission lines to get a constant global linear resistor network.

and each element's nonlinearity is isolated into the three local resistors as an elemental local system. TLM iteration exchanges information between all element systems and the global linear network until convergence, which is represented by the red arrows in Fig. 1.8.

Since the global resistor network is constant, the TLM decoupling technique only requires one global matrix factorization. Also, the local nonlinearity is isolated within each element's local Newton Rapson solving process, which means each element can be assigned to a CUDA thread, and the program can be massively paralleled at elemental-level, and the nonlinear solution is achieved without repetitive factorization/assembly of the global matrix.

1.5 Motivation of This Thesis

The above transmission line and NDD eliminate global NR iterations and is suitable for SIMD based GPUs. However, they are only developed for 2-D nodal elements in previous works, and 2-D representation and nodal FEM models have the following limitations:

- Power system devices are in 3-D forms. Representing a 3-D object with a 2-D model leads to geometry and field information loss. For example, 2-D models simplify and restrict all current flow and electric field in only one direction, which is obviously not true in reality.
- Nodal element formulations can only represent scalar physical fields, and the 3-D eddy current field formulation must be represented by 3 sets of nodal element scalar field for x,y,z directions respectively. This leads to significant numerical errors at sharp corners of the device geometry [18] Also, A penalty factor is always required in nodal formulations, which can affect the accuracy of the solution. If the device materials involve large permeability differences (i.e., the iron core and air), the numerical error becomes substantially large, regardless of the selection of the penalty factor.
- Many 2-D finite element cases have a total degree of freedoms less than 5000 [16], [17], which could be even less than a single GPU's Cuda cores. Thus, the computational resource cannot be fully used even with NDD and TLM schemes.

In contrast, 3-D edge elements [19] and vector basis functions, which is widely seen in 3D FEM analysis, do not have the above shortages. However, 3-D finite edge element models based on the above GPU-friendly transmission line decoupling and the DOF-level domain decomposition technology have never been implemented in the past.

Considering the above facts, it becomes paramount to explore new algorithms to implement the transmission line decoupling and DOF-level domain decomposition in 3D edges elements, especially for nonlinear transient electromagnetic analysis where a large amount of computational resource is needed.

1.6 Challenge in Extending From 2-D to 3-D

The 2-D FEM models ignore the change of magnetic potential change in 2 directions (x,y) of a 3-dimensional space. This significantly simplifies the full formulation in reality, and the 3-D edge FEM formulation is way more complex. For example, the 2-D model solves a scalar field distribution (only z direction), while the 3-D model deals with a vector field distribution. Thus, many challenges need to be solved in a 3-D FEM model for power system simulations.

$$\nabla^2 Az + \sigma \frac{\partial \mathrm{Az}}{\partial t} = Jz. \tag{1.1}$$

$$\nabla \times \nabla \times \nu \vec{A} + \sigma \frac{\partial \vec{A}}{\partial t} = \vec{J}, \tag{1.2}$$

(1.1) and (1.2) respectively presents the 2-D and 3-D physical equation of magnetic potential (which will be explained in the next chapters) in an eddy current field. For non-conducting materials such as air, the σ in 1.2 becomes 0. This means that the 3-D magnetic potential field does not have a unique solution, because the $\nabla \times \nabla \times$ operation can only define the curl of $\vec{A}$, but the divergence is not given, and a field can only be found when its curl and divergence are both defined. Therefore, a method must be found to gauge the divergence of $\vec{A}$ so that the 3-D global matrix can be factorized, which is not necessary in 2-D FEM, since the Az is fully defined.

In addition, the 3-D edge elements and 2-D nodal elements have entirely different interpolation functions. This means that the elemental field restrictions need to be re-derived and verified. Due to the huge difference between 2-D nodal and 3-D edge elemental restrictions, the 3-D elemental equations cannot be converted into 3 coupled nonlinear elemental resistors (Fig. 1.7) like the 2-D equations do. Also, because the mapping relation from magnetic potential to magnetic flux density ($\vec{B}$) is rather complex in 3-D, comparing with 2-D, the 3-D model cannot explicitly express the magnetic flux intensity and the permeability through the voltage on elemental equivalent resistors. The above facts leads to notable difficulties in the elemental local Newton Rapson iterations for 3-D edge elements.

1.7 Contributions of the Thesis

In this thesis, the transmission line decoupling technique is modified and successfully extended from 2-D nodal scalar elements (NSE) to 3-D edge vector elements (EVE) for power system EMT simulations. Challenges caused by the difference between 3-D EVE and 2-D NSE are successfully tackled. 3-D FE-discretized formulation for reduced magnetic vector potential is introduced. In contrast with 2-D triangular NSE, the 3-D EVE formulation is not full-ranked and a gauge is added to make a unique solvable nonlinear matrix system. An equivalent electrical circuit network is then extracted from the matrix system to facilitate the introduction of transmission lines. Next, this network is solved with elemental nonlinearity decoupled by the TLM technique. A new concept of scattering box is defined to abstract away elemental local nonlinearity in the form of 21 nonlinear resistors, compared with only 3 nonlinear resistors for 2-D triangular NSE. Also, the process of assigning value after partial Newton Rapson iterations is introduced to solve nonlinearity inside each element. The new process eliminates the need for an explicit function of $\vec{B}$ as a dependent variable of the voltage difference, which is essential in 2-D TLM FEM. Transmission lines are then used to separate the scattering boxes from the linear network, allowing

massive parallelism at the elemental level. This massive parallelized SIMD-friendly algorithm is verified by a GPU implementation, and the comparison between the proposed 3D-TLM-FEM scheme and $Comsol^{TM}$ indicates an excellent speedup of over 50 while a good precision (2%).

Also, the thesis extends the former effort of the 2-D NDDR scheme [16] further to the nonlinear 3-D edge elements. It shares the similar decentralized idea of applying domain-decomposition to each node and element. However, rather than solving a simple 2-D problem with scalar nodal unknowns, we achieved edge-level parallelism on the reduced magnetic vector potential for 3-D vector elements with much larger degree of freedoms. Different convergence and gauging behaviors are also explored, and an auto-gauging behavior is witnessed in the proposed algorithm, and its GPU implementation indicates a good accuracy (2%) and a decent speed-up of over 43 in comparison with $Comsol^{TM}$. The above explorations are based on space parallelism, either through 3-D elements or their edges. To fully increase parallelism, the thesis also adapts the time-parallel Parareal algorithm into 3-D finite element systems for power system EMT simulation, and time-space parallelism is achieved to fully utilize hardware resources. Furthermore, a new field-circuit coupling scheme is explored, which allows large eddy current and separate solution of the circuit and finite systems with high accuracy.

1.8 Thesis Outline

- **Chapter 1: Introduction** -The chapter introduces the background, motivation, and contribution of 3-D finite element models in power system electromagnetic transient simulations.

- **Chapter 2: 3-D finite element formulation for eddy current computation** -In this chapter, the 3-D vector potential formulation is derived from Maxwell's equation group for eddy current field computation, and edge tetrahedron finite element is applied to discretize the space and get a solvable matrix system, which can be parallelized at different levels.

- **Chapter 3: Transmission line decoupling for 3-D edge elements** -The chapter utilizes the transmission line numerical model to decouple the local nonlinearity in each element from the global finite element network. The global network stays unchanged during the entire computation process, thus only one-time factorization of the big-scale matrix is required, and local nonlinearity is assigned to different GPU cores to achieve element-level parallelism.

- **Chapter 4: Edge domain decomposition of 3-D edge elements** -The entire solution space is divided into many smaller sub-domains which only have one edge degree of freedom to form a light-weight nonlinear system. The sub-domains communicate

during a special iteration process and converge to a consistent solution over the entire space. Each sub-domain system is assigned to a GPU core to achieve edge-level parallelism. The widely applied time Parareal method is applied to the above space-parallelized 3-D finite element models to achieve parallelism at both space-time level.

- **Chapter 5: Conclusion** -The chapter gives the conclusion of the thesis and provides suggestions for future research.

2

3-D Finite Element Formulation for Eddy Current Field Computation

2.1 Introduction

In this chapter, the basic physical law of Maxwell's equation is applied for eddy current electromagnetic field computation. For the convenience of numerical solving, the 3-D vector potential is derived from Maxwell's equation to generate a continuous mathematical formulation over the entire space. Edge finite element is then applied to discretize the continuous formulation into a limited number of degree of freedoms (DOF). The chapter finally utilizes Galerkin's method to obtain a solvable matrix system for theses DOFs.

2.2 Quasi-Static Maxwell Equation and Formulation of Potential Function

Maxwell's equations are the basic laws of physics that describe electromagnetic phenomena in real life:

$$\left\{ \begin{array}{c} \nabla \times \vec{H} = \sigma \vec{E} + \vec{J}_e + \varepsilon \frac{\partial \vec{E}}{\partial \mathrm{t}} \\ \nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial \mathrm{t}} \\ \nabla \cdot \vec{B} = 0 \end{array} \right| \; entire\ solution\ domain. \tag{2.1}$$

The static case of the equation is used to solve static magnetic problems. For low-frequency electromagnetic field applications such as eddy current magnetic field analysis, capacitive effects such as displacement current and charge accumulation are ignored, thus

the ε related terms are deleted, resulting in the following quasi-static Maxwell's equation:

$$\left\{ \begin{array}{c} \nabla \times \vec{H} = \sigma \vec{E} + \vec{J}_e \\ \nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial \mathrm{t}} \\ \nabla \cdot \vec{B} = 0 \end{array} \right| entire\ solution\ domain, \tag{2.2}$$

where $\vec{J}_e$ is current defined inside conductors. The magnetic field induces different magnetizing currents in different materials, which in turn affects magnetic fields. This is described by the constitutive equation:

$$\vec{B} = \mu \vec{H} | entire\ solution\ domain. \tag{2.3}$$

The above equations indicate the following relation at the interface of two materials inside the solution domain:

$$\left\{ \begin{array}{c} \vec{B}_1^{\perp} = \vec{B}_2^{\perp} \ | interface\ of\ material\ 1\ and\ 2 \\ \vec{B}_1^{\parallel}/\mu_1 = \vec{B}_2^{\parallel}/\mu_2 \ | interface\ of\ material\ 1\ and\ 2 \end{array} \right. . \tag{2.4}$$

Together with proper boundary condition [20], $\vec{B}$ can be uniquely solved.

However, if $\vec{B}$ is set as the unknown variable to be solved, one can clearly see that the parallel component of the variable has to jump at every interface inside the solution domain. This is difficult for numerical computation, especially when multiple materials or nonlinearity is involved, since most interpolation functions of the finite element method are continuous, and it takes to much computation to specially deal with interfaces.

This problem can be solved by introducing Magnetic vector potential ($\vec{A}$). Considering $\vec{B}$ is divergence-free, it is replaced by $\nabla \times \vec{A}$ .It is possible to make $\nabla \times \vec{A}$ tangential continuous at material interfaces while setting $\vec{A}$ discontinuous. By substituting $\vec{B} = \nabla \times \vec{A}$ into the second equation in (2.2), one gets:

$$\nabla \times \left(\vec{E} + \frac{\partial \vec{A}}{\partial \mathrm{t}} \right) = 0, \tag{2.5}$$

which indicates:

$$\vec{E} = \overrightarrow{\nabla \varphi} - \frac{\partial \vec{A}}{\partial \mathrm{t}}. \tag{2.6}$$

By substituting (2.6) into the second equation in (2.2), and using 2.3 one gets:

$$\nabla \times \nabla \times \nu \vec{A} + \sigma \left(\frac{\partial \vec{A}}{\partial \mathrm{t}} - \nabla \varphi \right) = \vec{J}_e. \tag{2.7}$$

(2.7) have no unique solutions of $\vec{A}$ and φ because the relation between $\vec{A}$ and φ is not defined. One special $A - \varphi$ relation is letting $\vec{A}^* = \vec{A} + \int \overrightarrow{\nabla \varphi} \mathrm{dt}$ and therefore (2.7) becomes:

$$\nabla \times \nabla \times \nu \vec{A}^* + \sigma \frac{\partial \vec{A}^*}{\partial \mathrm{t}} - \vec{J}_e = 0, \tag{2.8}$$

which has a unique solution in conductive materials. This equation is the widely applied reduced vector potential formulation in eddy current magnetic field computation, and algorithms will be introduced in the thesis to solve the equation with state-of-art efficiency. For concise notation, sign $\vec{A}^*$ is replaced by $\vec{A}$ in the following part of this chapter.

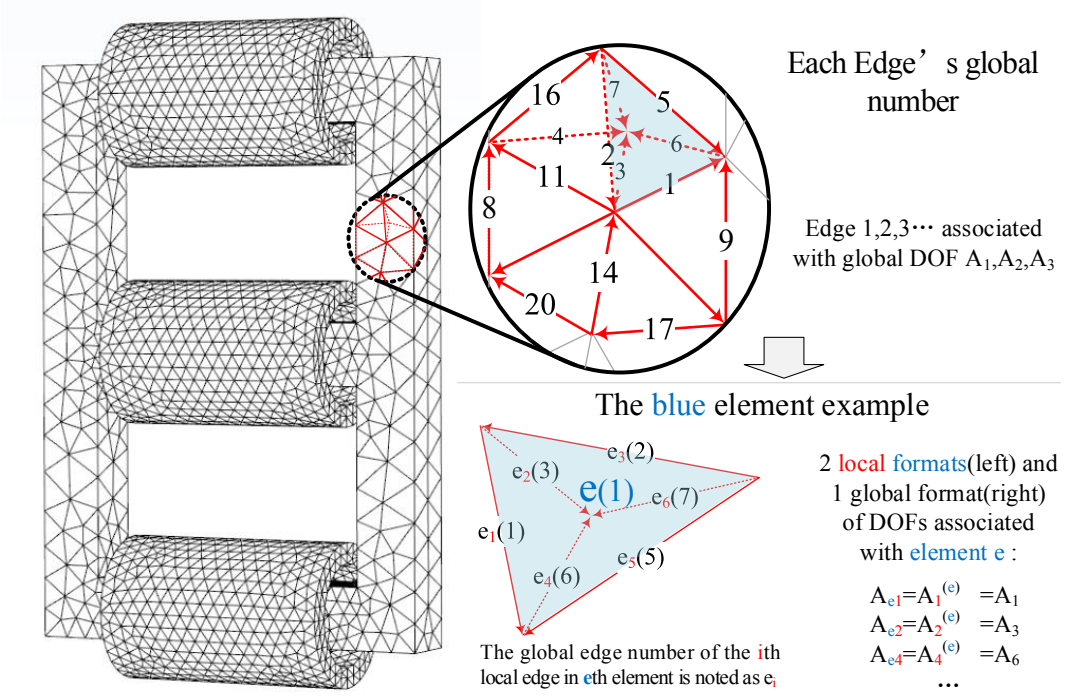


Figure 2.1: Solution space divided into many small tetrahedral elements and the local and global number notations for this thesis.

2.3 Gauge for Non-Conducting Materials

When the solution domain includes non-conduction regions, σ in (2.8) becomes 0 for such region. Inside such region, (2.8) will have infinite sets of solutions: (2.8) loses restriction to the divergence of $\vec{A}$ and only gives information for rotation of $\vec{A}$, and a vector field cannot be uniquely defined unless both its rotation and divergence are known. Different regulations can be added to restrict the divergence of $\vec{A}$. The regulations are called gauges and will be introduced later in the next chapters, since gauges are embedded into the computation algorithms.

2.4 Finite Element Discretization of The Reduced Magnetic Vector Potential Formulation

The electromagnetic field distribution can be restored from the vector potential $\vec{A}$ after the solution of (2.1) through the relation between magnetic flux density and vector potential:

$$\vec{B} = \nabla \times \vec{A}. \tag{2.9}$$

However, (2.1) is in a continuous form. This means (2.1) still needs further discretization into a limited number of degree of freedoms (DOFs) for numerical solutions.

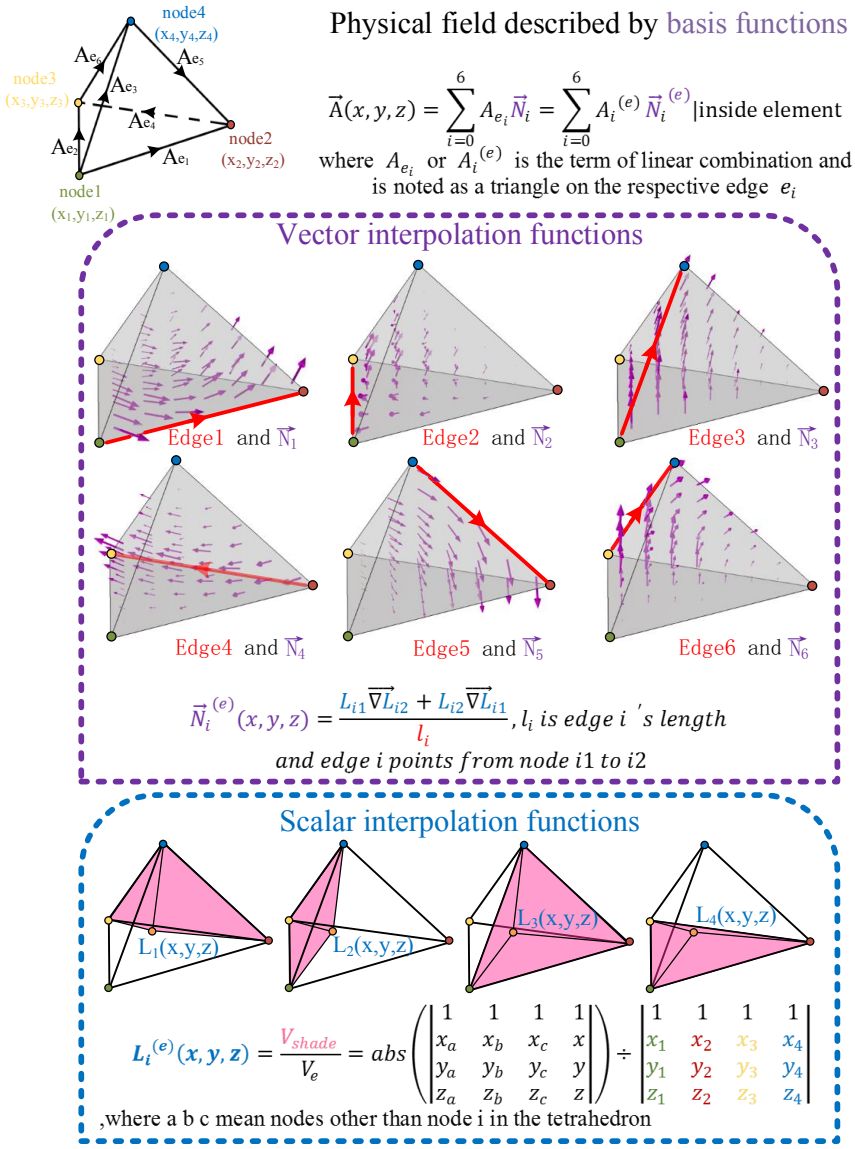


Figure 2.2: Interpolation functions for 3-D tetrahedron elements and $\vec{A}$ field described by edge vector interpolation functions in an element. Note: upper index (e) means variables or functions in eth element.

To achieve this, the finite element method divides the whole solution domain into many smaller tetrahedral elements and edges between the tetrahedrons ((Fig. 2.1)). Each edge has its own direction and one DOF (A_j related to edge number j) that is equal to the projection of $\vec{A}$ onto its direction. For eth element, e_i represents the global number of ith edge in the eth element (i $= 1\tilde{6}$), and within each element, $\vec{A}$ field is represented through a linear combination of know pattern fields (interpolation functions). As is shown in Fig. 2.2, the interpolation functions ($\overrightarrow{N_i}$) are fully determined by the coordinates of the 4 elemental nodes (known), and $\overrightarrow{N_i}$'s projection is 1 on edge i's direction while 0 on all other edges. This means that the 6 involved linear combination terms ($A_{e_1}, A_{e_2}, A_{e_3}$, or $A_1^{(e)}$,

$A_2^{(e)}$, $A_3^{(e)}\dots$) of each edge is exactly the DOF on the respective edge. To find the value of these DOFs (namely, the solution of FEM problem), one can construct local restrictions between the 6 involved edge DOFs for every single element, and add these local equations to a global matrix system. The matrix system can then be solved for all edge DOFs.

The Galerkin weighted residual method can be used to obtain elemental restrictions. The residual is defined by discretizing the left-hand side of (2.8) through the upper equation in Fig. 2.2. :

$$\nabla \times \nabla \times \nu \sum_{i=1}^{6} A_{e_i} \cdot \overrightarrow{N_i}^{(e)} + \sigma \frac{\partial \sum_{i=1}^{6} A_{e_i} \cdot \overrightarrow{N_i}^{(e)}}{\partial \mathrm{t}} - \vec{J}_e = 0. \tag{2.10}$$

This is a linear combination of A_{e_i}. The residual should always be 0 over the entire space. Thus, one can multiply any function with the residual, and force the result into 0 in order to find the degree of freedoms (A_{e_i}). If one multiplies the residual with 6 weighing functions (chosen to be Nj), the following equation can be obtained:

$$\sum_{i=1}^{6} A_{e_i} \cdot \int_{Ve} \overrightarrow{N_j}^{(e)} \cdot \left(\nu_e \nabla \times \nabla \times \overrightarrow{N_i}^{(e)} + \sigma \frac{\partial \overrightarrow{N_i}^{(e)}}{\partial \mathrm{t}} \right) dV \; = \int_{Ve} \overrightarrow{N_j}^{(e)} \cdot \vec{J}_e dV, \tag{2.11}$$

where Ve means the space volume of eth element. Due to the below vector identity formulation:

$$\vec{a} \cdot \left(\nabla \times \left(\nabla \times \vec{b} \right) \right) = (\nabla \times \vec{a}) \cdot \left(\nabla \times \vec{b} \right) - \nabla \cdot \left(\vec{a} \times \left(\nabla \times \vec{b} \right) \right), \tag{2.12}$$

and the Gauss's divergence theorem:

$$\int_{Ve} \nabla \cdot \vec{a} \; dV = \int_{Se} \vec{a} \cdot \overrightarrow{ds}, \tag{2.13}$$

(2.11) becomes equivalent to the 6 following equations:

$$\sum_{i=1}^{6} A_{e_i} \cdot \int_{Ve} \left(\nu_e \; \left(\nabla \times \overrightarrow{N_j}^{(e)} \right) \cdot \left(\nabla \times \overrightarrow{N_i}^{(e)} \right) + \sigma \frac{\partial \overrightarrow{N_i}^{(e)} \cdot \overrightarrow{N_j}^{(e)}}{\partial \mathrm{t}} \right) dV \; = \int_{Ve} \overrightarrow{N_j}^{(e)} \cdot \vec{J}_e dV, \tag{2.14}$$

where $j = 1\tilde{}6$. For (2.14), the 6*6 matrix elemental discretized formulation is obtained:

$$\left[K_{e_i e_j}\right]_{6\times 6} \cdot \left[A_{e_j}\right]_{6\times 1} + \left[D_{e_i e_j}\right]_{6\times 6} \cdot \frac{\partial}{\partial t} \left[A_{e_j}\right]_{6\times 1} = I \left[f_{e_i}\right]_{6\times 1} + \left[\tau_{e_i}\right]_{6\times 1}, \tag{2.15}$$

where

$$K_{e_i e_j} = \int_{Ve} \nu_e \left(\nabla \times \overrightarrow{N_i}^{(e)} \right) \cdot \left(\nabla \times \overrightarrow{N_j}^{(e)} \right) \mathrm{dV}, \mathrm{i}, \mathrm{j} = 1\tilde{}6, \mathrm{j} = 1\tilde{}6, \tag{2.16}$$

$$D_{e_i e_j} = \int_{Ve} \sigma_e \overrightarrow{N_i}^{(e)} \cdot \overrightarrow{N_j}^{(e)} \mathrm{dV}, \mathrm{i} = 1\tilde{}6, \mathrm{j} = 1\tilde{}6, \tag{2.17}$$

$$f_{e_i} = \int_{Ve} \overrightarrow{N_i}^{(e)} \cdot \overrightarrow{J_{unit}} dV, \tag{2.18}$$

$$\tau_{e_i} = \int_{Se} \left(\nu \nabla \times \vec{A} \right) \times \overrightarrow{ds} \cdot \overrightarrow{N_i}^{(e)}, \tag{2.19}$$

and Ve is the volume, Se is the surface boundary of eth element, $\overrightarrow{J_{Unit}}$ is the current density distribution at 1 Amp current in the coil of the power device. I is the known current in the coil which changes with time

The time derivative in (2.15) can also be discretized using Backward Euler method:

$$\left[M_{e_i e_j}\right]_{6\times 6} \cdot \left[{A_{e_j}}^t\right]_{6\times 1} = \left[b_{e_i}\right]_{6\times 1} + \left[\tau_{e_i}\right]_{6\times 1}, \tag{2.20}$$

where

$$\left[b_{e_i}\right]_{6\times 1} = I \left[f_{e_i}\right]_{6\times 1} + \frac{\left[D_{e_i e_j}\right]_{6\times 6}}{\Delta t} \cdot \left[{A_{e_j}}^{t-1}\right]_{6\times 1}, \tag{2.21}$$

$$\left[M_{e_i e_j}\right]_{6\times 6} = \left[K_{e_i e_j}\right]_{6\times 6} + \left[D_{e_i e_j}\right]_{6\times 6}/\Delta t, \tag{2.22}$$

and ${A_{e_j}}^t$ is the unknowns to solve at current time step, and ${A_{e_j}}^{t-1}$ is A_{e_j} at the previous time step. , and $\left[M_{e_i e_j}\right]_{6\times 6}$, $\left[K_{e_i e_j}\right]_{6\times 6}$, $\left[D_{e_i e_j}\right]_{6\times 6}$are noted as M_e, K_e, D_e for simplicity.

To achieve a solution over the entire domain for all elements, the local restrictions of 6 edge DOFs in (2.20) related to each element need to be assembled into a global system. During this process, all elements 'talk' with each other at surfaces between different elements with respect to Maxwell equations [21]: 1. Continuity of tangential $\vec{A}$ is automatically ensured by the edge-based DOF discretization format. As a result, normal $\vec{B}$ is continuous between elements and the upper equation in (2.4) are satisfied. 2. On a triangular surface shared by 2 elements, the τ terms in the 2 elements are opposite to each other for the same edge, which leads to the continuity of tangential H and consistency with the lower equation in (2.4). For ith edge, this relation can be expressed as:

$$\sum_e \tau_{e_k} = 0 | e_k = i, \mathrm{k} = 1\tilde{6}, \tag{2.23}$$

where e represents all elements sharing edge i.

During assembly, local restrictions (2.9) for all elements are directly added together to make the global matrix equation below for all global edge DOFs, and the surface integration term does not appear in the global equation. Thus (2.1) and (2.9) are simultaneously satisfied.

$$M \cdot A^t = \left(K + \frac{D}{dt} \right) \cdot A^t = b, \tag{2.24}$$

where $K = \sum_{e=1}^{E} K_e$, $D = \sum_{e=1}^{E} D_e$, $\mathrm{b} = f \cdot {I_k}^t + \frac{D}{dt} \cdot A^{t-1}$, $f = \sum_{e=1}^{E} f_e$, and E is the total number of elements.

It can be seen that the matrix M in (2.24) is nonlinear since ν_e in (2.16) is dependent on A_{e_j}. Such a DOF-level-sized nonlinear system leads to excessive computation burden. The

following chapters will discuss how to efficiently solve the nonlinear global matrix system with parallel computation hardware, and a little modification to the notation of the above matrix system is made in order to concisely adapt to the specific algorithms.

2.5 Summary

This chapter explained the derivation process of the finite element matrix system from the physical control equations. The Maxwell's equations were first simplified into a reduced vector magnetic potential formulation. The potential formulation was then discretized using finite element representation of elemental local restrictions, which were assembled into a global matrix system to be solved. The chapter also explained the rank deficiency of the matrix system due to non-conducting materials.

3

Transmission Line Decoupling for 3-D Edge Elements

3.1 Introduction

In this chapter, the transmission line modeling method is combined with 3-D finite edge elements to handle nonlinearity with parallelism at the tetrahedron element level, and the background, derivation, and effectiveness are discussed for the 3-D transmission-line-decoupled finite element model.

The finite element method has been widely applied for field calculation in electromagnetic apparatus such as transformer, electrical motor, and power inductor, where complex geometry and nonlinearity stemming from ferromagnetic materials are always encountered. However, traditional FEM suffers from severe computation burden due to repetitive assembly and factorization of the large-scale Jacobian global matrix at each Newton Rapson (NR) iteration step.

During the last decade, high-performance computation hardware has witnessed a dramatic increase in the number of computational units. This encourages researchers to accelerate the finite element computation with the help of multi-core CPUs or many-core GPUs.

Under such circumstances, the TLM, which was introduced to nonlinear circuit network [22], [23] and 2D FEM [24], [25] decades ago, has witnessed increasing attention recently [17], [26]. When a nonlinear system is solved by TLM, the update happens only inside the system vector in each iteration step. This indicates that the system matrix remains the same during the whole solution process and only one-time matrix factorization is necessary. In addition, the parallelism of the TLM-FEM is high because elemental local nonlinearity is separated by transmission lines. By utilizing these properties, the speed of

TLM-FEM is fast enough even for real-time simulation [17].

However, all of the above TLM-FEM applications are based on 2-D triangle elements, while in reality, the geometry of an electromagnetic model is in 3-D form but the 3-D-TLM-FEM implementation is never conducted. Moreover, the above FEM-TLM was only implemented for nodal elements in the scalar form, however, nodal elements result in a large inaccuracy at sharp corners of geometry and when materials have large permeability differences, the numerical error becomes substantial [20], [18]. On the contrary, edge elements [19] and vector basis functions, which is widely seen and has been utilized for network equivalence [27], [28] in 3-D FEM analysis, do not have such problem. Considering the above facts, it becomes paramount to explore new algorithms to implement transmission line modeling in 3-D edges elements, especially for nonlinear transient electromagnetic analysis where a large amount of computational resources are needed.

In this chapter, the transmission line decoupling technique is modified and extended from 2-D nodal scalar elements (NSE) to 3-D edge vector elements (EVE) in nonlinear electromagnetic field problems. Challenges caused by the difference between 3-D EVE and 2-D NSE are successfully tackled. Firstly, 3-D finite element discretized formulation for reduced magnetic vector potential (MVP) is introduced. In contrast with 2-D triangular NSE, 3-D EVE formulation is not full-ranked and a gauge is added to make a unique solvable nonlinear matrix system.

An equivalent electrical circuit network (rather than a magnetic equivalent circuit network) is then extracted from the matrix system to facilitate the introduction of transmission lines. Next, this network is solved with elemental nonlinearity decoupled by the TLM technique. A new concept of scattering box is defined to abstract away elemental local nonlinearity in the form of 21 nonlinear resistors, compared with only 3 nonlinear resistors for 2-D triangular NSE. Also, the process of assigning value after partial NR iterations is introduced to solve nonlinearity inside each element. The new process eliminates the need for an explicit function of $\vec{B}$ as a dependent variable of the voltage difference, which is essential in 2-D TLM FEM. Transmission lines are then used to separate the scattering boxes from the linear network, allowing massive parallelism at the elemental level.

This chapter is arranged as follows: 3.2 introduces the gauged eddy current formulation based on reduced MVP, which is discretized by edge element interpolation functions in Section 3.3. 3.4 and 3.5 describe the massively parallelized TLM-FEM scheme of the discretized formulation. In 3.6, the massive parallelism is verified by a GPU implementation, and a comparison between the proposed 3-D TLM-FEM scheme and $Comsol^{TM}$ indicates an excellent speedup of over 50 while a good precision (2%). Finally, Section 3.7 and 3.8 give a discussion and summary of the chapter.

3.2 FEM Formulation For Eddy Current Analysis

3.2.1 The Reduced Magnetic Potential Formulation

As is introduced in chapter 2, the quasi-static Maxwell's equations give restriction of magnetic flux density ($\vec{B}$) in eddy current analysis. Through Maxwell's equations, it is obvious that $\vec{B}$ jumps at the interface between different materials, which causes inconvenience for numerical computation [29]. To deal with such a problem, the well-known $\vec{A}$-φ formulation was introduced and $\vec{A}$,φ can be made continuous between materials [29]. The $\vec{A}$-φ formulation can be simplified to the reduced $\vec{A}$ formulation (3.1) when edge element is used, which allows perpendicular jump of $\vec{A}$ between different conductivities:

$$\nabla \times (\upsilon \nabla \times \vec{A}) + \sigma \frac{\partial \vec{A}}{\partial t} = \vec{J}_e, \tag{3.1}$$

where υ is the field-dependent reluctivity; σ is the electrical conductivity; $\vec{J}_e$ is the impressed current density.

3.2.2 Gauge for Non-Conducting region

The system in (3.1) is uniquely solvable if all solution domain are conductors. However, when solution domain (Ω_{all}) includes non-conduction region (Ω_n), (3.1) does not have unique solution because it loses restriction to the divergence of $\vec{A}$ in Ω_n . To get a unique solution, an innovative gauge [30], [31] can be added by introducing a dummy variable χ in Ω_n, which results in the following:

$$\begin{cases} \nabla \times (\upsilon \nabla \times \vec{A}) + \sigma \frac{\partial \vec{A}}{\partial t} - \nabla \chi = \vec{J}_e, \\ \nabla \cdot \vec{A} + \chi = 0. \end{cases} \tag{3.2}$$

Suppose $\vec{J}_e$ is divergence-free, by taking divergence to both sides of the first equation in (3.2), one finds that χ satisfies the Laplace equation:

$$\nabla^2 \chi = 0 | \ \ \Omega_n. \tag{3.3}$$

By forcing χ to be zero at the outter boundary of Ω_n:

$$\chi = 0 | \ \ \Gamma_n, \tag{3.4}$$

where Γ_n is the boundary of Ω_n, (3.3) becomes a simple boundary value problem and the solution is 0 in Ω_n. Now that χ is 0, considering second equation of (3.2), $\nabla \cdot \vec{A}$ becomes 0. Therefore, Coulomb gauge is applied to $\vec{A}$ to get a unique solution. For simplicity, the homogeneous Dirichlet boundary is imposed to $\vec{A}$ in this work:

$$\vec{A}^{\parallel} = 0 | \ \ \Gamma_{all}, \tag{3.5}$$

where Γ_{all} is the boundary of Ω_{all}. Equations (3.2), (3.4), (3.5) form the formulation used for eddy current analysis in this chapter.

3.3 Finite Element Method And Discretized Formulation

3.3.1 The Magnetic Vector Potential Field

Finite element method can be used to solve the above vector-and-scalar-hybrid formulation. It simplifies the problem by representing the unknown field with a limited degree of freedom. The solution domain is divided into many subdomains (elements). Within each element, the unknown field is a linear combination of known pattern fields (interpolation functions fully determined by the coordinate of the element, shown in Fig. 2.2. The terms ($A_1^{(e)}$, $A_2^{(e)}$, $A_3^{(e)}\dots$) of this linear combination for every element are the degree of freedom to be solved.

Galerkin method can be used to find those terms. For the method, the weight function is the same as the interpolation function. The $\vec{A}$ and χ in 3.2 can be discretized to obtain the numerical residual by the upper equation in Fig. 2.2 and the following scalar field representation:

$$\chi\left(x,y,z\right)=\sum_{i=0}^{4}\chi_i {L_i}^{(e)}|\text{inside element e} \tag{3.6}$$

For concise notation, the element number index e in chapter 2 is omitted in the following derivations ($A_i^{(e)}$ or A_{e_i} changes to A_i, $K_{e_ie_j}$ changes to K_{ij}). If one use vector identities and force integration of the product of weight function and the residual to be zero within one element, the elemental discretized formulation is obtained as follows:

$$\int_{Se}(\upsilon\nabla\times\vec{A})\times\vec{ds}\cdot\vec{N}_i+\int_{Ve}(\upsilon\nabla\times\vec{A})\cdot\nabla\times\vec{N}_i dV + \sigma\frac{\partial}{\partial t}\int_{Ve}\vec{N}_i\cdot\vec{A}dV-\int_{Ve}\vec{N}_i\cdot\nabla\vec{\chi}dV-\int_{Ve}\vec{N}_i\cdot\vec{J}_e dV=0, \tag{3.7}$$

$$\int_{Se}L_i\vec{A}\vec{ds}+\int_{Ve}\vec{A}\cdot\nabla\vec{L}_i dV-\int_{Ve}L_i\chi dV=0, \tag{3.8}$$

where Se is the surface boundary of the element and i is an integer from 1 to 6. By substituting 2 FEM-discretized field expressions at the top of Fig. 2.2 into (3.7)-(3.8) and ignoring the first term in (3.7)-(3.8), the matrix form element discretized formulation is obtained:

$$\begin{bmatrix}[\widehat{AA}_{ij}]_{6\times6} & [\widehat{A\chi}_{ij}]_{6\times4}\\ [\widehat{\chi A}_{ij}]_{4\times6} & [\widehat{\chi\chi}_{ij}]_{4\times4}\end{bmatrix}\begin{bmatrix}[A_j]_{6\times1}\\ [\chi_j]_{4\times1}\end{bmatrix}+ \sigma\frac{\partial}{\partial t}\begin{bmatrix}[\widehat{CC}_{ij}]_{6\times6} & [0]_{6\times4}\\ [0]_{4\times6} & [0]_{4\times4}\end{bmatrix}\begin{bmatrix}[A_j]_{6\times6}\\ [0]_{4\times6}\end{bmatrix}=\begin{bmatrix}[b_i]_{6\times6}\\ [0]_{4\times6}\end{bmatrix}, \tag{3.9}$$

where

$$\widehat{AA}_{ij}=\int_{Ve}(\upsilon\nabla\times\vec{N}_i)\cdot(\nabla\times\vec{N}_j)dV, i,j\in[1,6], \tag{3.10}$$

$$\widehat{A\chi}_{ij}=\int_{Ve}\vec{N}_i\cdot\nabla\vec{L}_j dV, i,j\in[1,6], \tag{3.11}$$

$$\widehat{\chi A}_{ij} = \int_{Ve} \nabla\vec{L}_i \cdot \vec{N}_j dV, i, j \in [1, 6], \tag{3.12}$$

$$\widehat{\chi\chi}_{ij} = \int_{Ve} L_i L_j dV, i, j \in [1, 6], \tag{3.13}$$

$$\widehat{CC}_{ij} = \int_{Ve} \vec{N}_i \cdot \vec{N}_j dV, i, j \in [1, 6], \tag{3.14}$$

$$b_i = \int_{Ve} \vec{N}_i \cdot \vec{J}_e dV, i, j \in [1, 6]. \tag{3.15}$$

However, (3.9) only indicates the relation between edge/node unknowns that belong to the corresponding element. In the assembling phase, the relation between all unknowns of the solution domain is constructed by adding the contribution of every element to a global matrix. The surface integration in (3.7)-(3.8) is not shown in (3.9). In fact, they are canceled out during the assembling to ensure the continuity of tangential $\upsilon\nabla \times \vec{A}$ and the continuity of perpendicular $\vec{A}$ at the interface between every element. As a result, Coulomb gauge is automatically satisfied and every element can have different υ with almost no computational expense. Note that A_i is not the value of x, y, or z component of magnetic potential on the corresponding edge i. In fact, A_i is terms of the edge's interpolation function. The value of A_i is the value of the projection of magnetic potential to the edge's direction. According to the definition of edge interpolation function, $\vec{B}$ can be expressed by A_i:

$$\vec{B} = \nabla \times \vec{A} = \sum_{i=1}^{6} A_i \nabla \times \vec{N}_i = \sum_{i=1}^{6} A_i \frac{2\nabla\vec{L}_{i1} \times \nabla\vec{L}_{i2}}{l_i}. \tag{3.16}$$

3.3.2 The Excitation Current Field

The excitation current field is calculated by solving a 3D static current conservation problem:

$$\begin{cases} \nabla \cdot \vec{J}_e = -\nabla^2 \sigma_{cl} \varphi = 0 |\ \ \Omega_{cl}, \\ \varphi = 0 |\ \ \Gamma_{Gd}, \\ \varphi = V(t) |\ \ \Gamma_{Vs}, \\ \vec{n} \cdot \vec{J}_e = \vec{n} \cdot \vec{\nabla\varphi} |\ \ \Omega_{cl}, \end{cases} \tag{3.17}$$

where Ω_{cl} is the coil domain, Γ_{Gd}, Γ_{Vs} are the boundary of coil surface, ground, and excitation voltage. V(t) is the voltage used to calculate time-variant excitation current field, and σ_{cl} is the material conductivity of the coil. The coil domain is discretized by tetrahedrons and the scalar basis function L_i shown in Fig. 2.2. Similarly, (3.17) is discretized with Galerkin method. The voltage value at each tetrahedron vertex is then solved, and the current inside each element is found by $\sigma_{cl}\vec{\nabla\varphi}$.

3.4 Solving the Nonlinear System

3.4.1 TLM Method for FEM Analysis

The nonlinear system assembled by (3.9) is usually solved by NR or quasi-Newton method. However, these methods require updating of the global matrix, which slows down the computation. The transmission line method, in contrast, does not have such a problem.

The transmission line method was already implemented to handle nonlinearity, despite its full-wave physical essence. Although the TLM originates from the Huygens' wave propagation model and was used to solve full-wave problems [23], due to the delaying and isolating function of the transmission line, TLM was successfully extended to solve nonlinear problems, such as the solution of a nonlinear electrical circuit network [22]. By converting 2D FEM equations into a nonlinear electrical circuit equivalence, the TLM method was further implemented for 2D magnetic field analysis [24].

To stay consistent with previous TLM-FEM works, we opted to represent the 3D FEM equations with a similar nonlinear equivalent circuit model. According to the circuit analysis theory [32], the symmetric matrix system in (3.9) can be represented by a circuit network. The RHS corresponds to node injection currents and the unknowns correspond to circuit node voltage. As to the first matrix, the negative value of matrix element ij corresponds to the conductance between nodes i and j. Sum of the first 6 elements in row i corresponds to the nonlinear conductance of node i to ground and sum of the last 4 elements corresponds to the linear ground conductance of node i. Similarly, the second matrix can be represented by a linear capacitor network. Since the two admittance matrixes in (3.9) are added together, the resistive and capacitive networks are, according to circuit analysis theory, parallel connected to the same circuit topology, which is shown in the next part.

Nonlinear resistors in the circuit network are separated from the linear circuit by transmission lines with arbitrary characteristic impedance Z_c. The linear circuit only sees linear Z_c and 'communicates' with nonlinear resistors by traveling waves on transmission lines. After reflecting between the two line terminals many times, the waves reach steady values. In this way, nonlinearity is replaced by equivalent current sources and the admittance matrix stays the same during each TLM iteration.

It is worth mentioning that υ is constant inside one element, which implies the coupling of all nonlinear resistors inside one element. These resistors should be treated as a small subsystem separated by transmission lines for each element. The subsystem is defined as a 'scattering box' in this chapter. Since these small subsystems (scattering boxes) for each element are separated by transmission lines, it is possible to solve each subsystem independently of others.

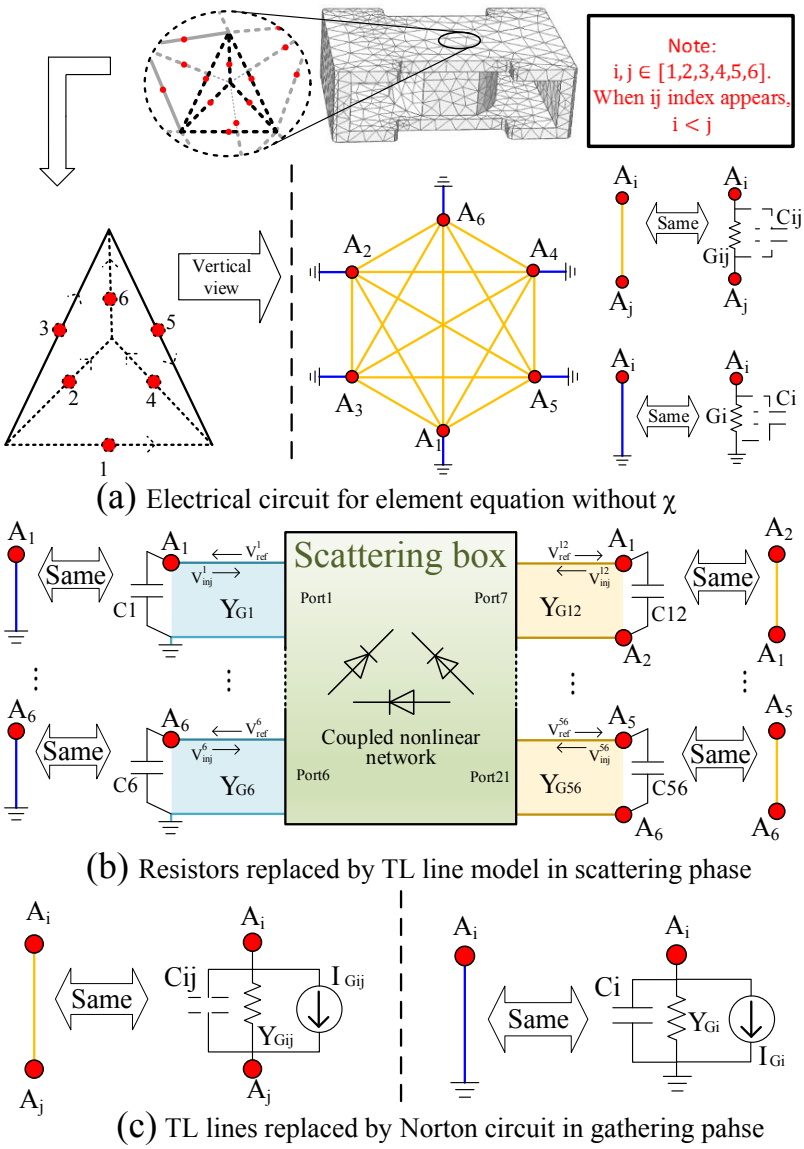


Figure 3.1: TLM technique for nonlinear system in (3.9) without χ.

3.4.2 Elemental Electrical Circuit and TLM Process

In (3.9), matrix elements involving χ do not include υ and they correspond to linear resistors. For simplicity, they are not shown in the elemental circuit model. Fig. 3.1a shows the nonlinear resistors of the equivalent circuit model of submatrix $[\widehat{AA}_{ij}]_{6\times6}$, $\sigma[\widehat{CC}_{ij}]_{6\times6}$ in (3.9). The 6 red dots are the circuit nodes and the voltage of the nodes (A_i) corresponds to the value of projection of $\vec{A}$ field to the corresponding edge direction. Yellow lines are the branches between two nodes and blue lines are the branch between each node and ground. The circuit component of each branch is shown on the right side with the following values:

$$Gij = -\int_{Ve} \upsilon_e(\nabla \times \vec{N}_i) \cdot (\nabla \times \vec{N}_j) dV \quad i,j \in [1,6], i \neq j, \tag{3.18}$$

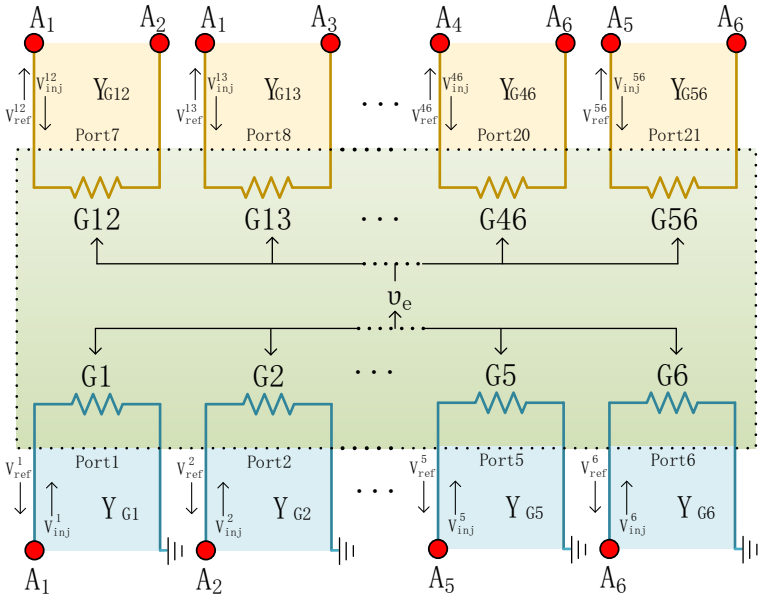


Figure 3.2: Inside scattering box: coupled elemental nonlinear network for scattering phase.

$$Gi = \sum_{j=1}^{6} \int_{Ve} \upsilon_e (\nabla \times \vec{N}_i) \cdot (\nabla \times \vec{N}_j) dV, \quad i, j \in [1, 6], \tag{3.19}$$

$$Cij = -\sigma \widehat{CC}_{ij}, \quad Ci = \sum_{j=1}^{6} \widehat{CC}_{ij} \quad i, j \in [1, 6], \tag{3.20}$$

where υ_e is the unknown reluctivity determined by A_i. υ_e is to be solved by NR iteration inside the scattering box.

The TLM process is divided into 2 major phases: scattering and gathering. In the scattering phase shown in Fig. 3.1b, the nonlinear resistors are replaced by transmission lines and a scattering box. The transmission line characteristic conductance is given by:

$$Y_{Gij} = -\int_{Ve} \upsilon_e^g (\nabla \times \vec{N}_i) \cdot (\nabla \times \vec{N}_j) dV \quad i, j \in [1, 6], i \neq j, \tag{3.21}$$

$$Y_{Gi} = \sum_{j=1}^{6} \int_{Ve} \upsilon_e^g (\nabla \times \vec{N}_i) \cdot (\nabla \times \vec{N}_j) dV, \quad i, j \in [1, 6], \tag{3.22}$$

where υ_e^g is the guessed elemental reluctivity value before computation and should be as close as possible to the final solution of υ_e.

In this phase, information of the global circuit is fed to elemental nonlinear resistors through transmission lines. Injection waves affected by nodal voltages (A_i) and branch voltages (A_i-A_j) enter the scattering box (Fig. 3.2). Since each scattering box only sees

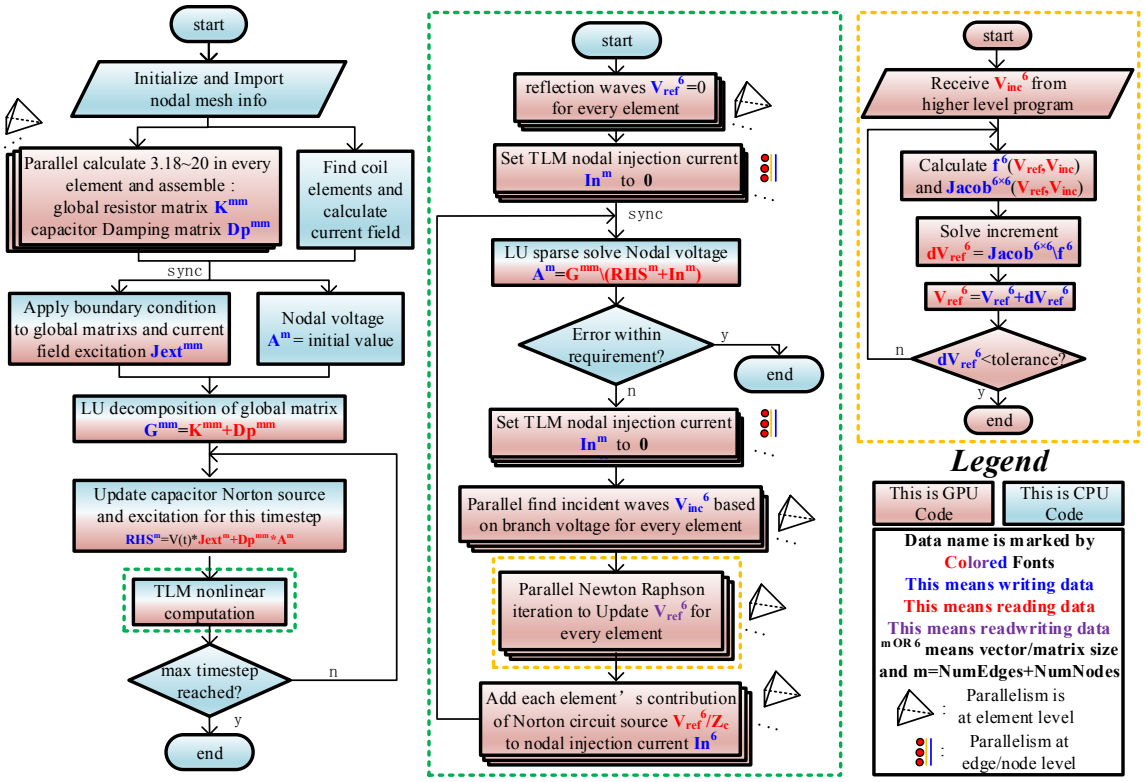


Figure 3.3: Detailed massively parallel implementation of the 3D-TLM-FEM method.

transmission lines and incident waves rather than other scattering boxes, the solution process inside each scattering box is independent of other scattering boxes. Inside each individual scattering box, incident wave at one port is scattered to all ports according to the coupling relation of nonlinear resistors. After scattering, waves leave the box and becomes reflection waves. For example, V_{inj}^1 enters port 1 and it is distributed and guided towards ports 1 to 21 based on the relation between G1 and other conductance. All incident waves are recombined to form reflection waves. Since v_e is determined by (A_i) rather than (A_i-A_j), only blue ports participate in the NR iteration. The nonlinear relation between V_{inj} and V_{ref} is the following:

$$Gi(V_{inj}^i + V_{ref}^i) - Y_{Gi}(V_{inj}^i - V_{ref}^i) = 0 \quad i \in [1, 6], \tag{3.23}$$

where Y_{Gi} are guessed (fixed) but Gi are functions of $V_{inj}^i + V_{ref}^i$. These functions are determined by (3.16) and $B - v/B - H$ curve. To calculate this 6×6 system, NR iteration is used and the following equation yields the Jacobian matrix:

$$\frac{\partial Gi}{\partial V_{ref}^j} = \frac{\partial Gi}{\partial B^2}\frac{\partial B^2}{\partial V_{ref}^j} \quad i, j \in [1, 6]. \tag{3.24}$$

After the 6×6 NR iteration, v_e is found, and v_e is used to calculate the Gij at port 7~21, and

the reflection waves for port 7∼21 can be found by:

$$V_{ref}^{ij} = V_{inj}^{ij} \frac{Gij - Y_{Gij}}{Gij + Y_{Gij}} \;\; i, j \in [1, 6] \tag{3.25}$$

The above local NR solving process does not involve nodal voltage difference (Ai-Aj). In other words, the solution process does not need an explicit function of $\vec{B}$ as a dependent variable of the voltage difference. Therefore, an extra transformation is unnecessary for (3.16), which might be extremely difficult.

During the gathering phase (Fig. 3.1c), the reflected waves carrying the information from elemental nonlinear resistors return to the global circuit network. Transmission lines and reflected waves are replaced by the Norton model, and node voltages are then updated for the global circuit network and these voltages are used to generate injection waves for the next iteration. It is worth mentioning that the circuit network in such phase is a massive network assembled from all elemental circuits in the mesh.

As is explained above, each TLM iteration involves many independent parallelizable solutions for scattering boxes during the scattering phase and one solution of the fixed global admittance matrix.

3.4.3 Time Discretization for Capacitors

As is shown in Fig. 3.1, there are capacitors in parallel with nonlinear resistors. The capacitors involve time derivatives that can be replaced by a resistor and a history current source by different numerical methods (backward Euler rule used in this work). The admittance and current source are given by:

$$Y_{Cij} = Cij/\Delta t \;\; Y_{Ci} = Ci/\Delta t i, j \in [1, 6], \tag{3.26}$$

$$I_{Cij} = Y_{Cij}(A_i^{(n-1)} - A_j^{(n-1)}) \;\; i, j \in [1, 6], \tag{3.27}$$

$$I_{Ci}^{(n)} = Y_{Cij} A_i^{(n-1)} \;\; i, j \in [1, 6], \tag{3.28}$$

where Δt is the time-step, upper index (n) means current time-step and $(n-1)$ means previous time-step.

3.5 Program for Massively Parallel Architecture

As is mentioned above, the scale of NR iteration in the TLM method is small (6*6). More importantly, the NR iteration of different elements is independent and thus can be massively parallelized. In addition, TLM iterations do not change the admittance matrix, thus only one time of admittance matrix inversion is needed for the whole solution process. These properties make it suitable to run the proposed 3D-TLM edge finite element method on massively parallelized architectures such as the GPU. A Cuda C program is developed and the flowchart is shown in Fig. 3.3 for GPU implementation.

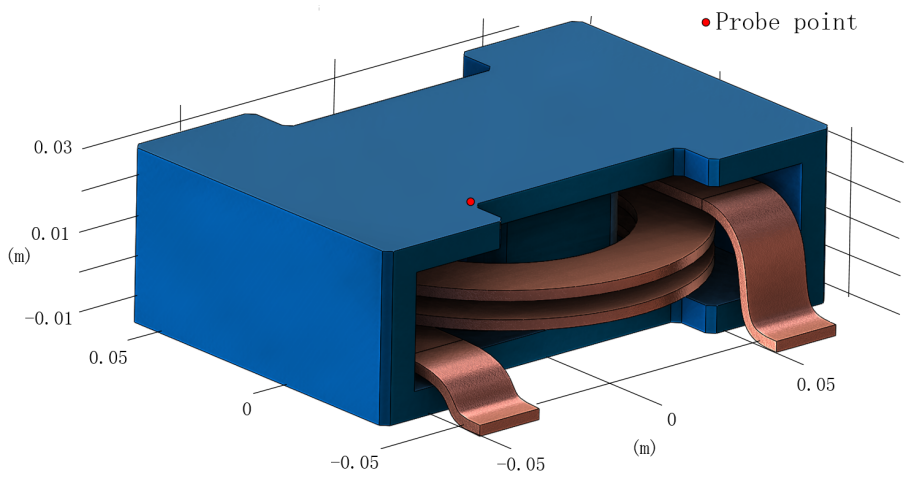


Figure 3.4: Dimensions of studied power inductor.

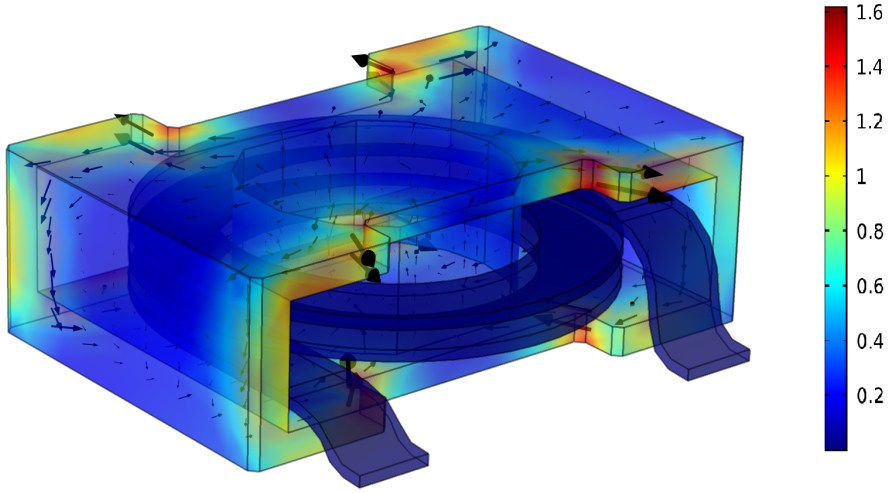


Figure 3.5: Magnetic flux density distribution (module and vector) at 0.00667s.

3.6 Case Study

To demonstrate the precision and efficiency of the 3D-TLM-FEM method, a power inductor shown in Fig. 3.4 is studied in comparison with $Comsol^{TM}$. The blue iron core has a size of 0.15* 0.1*0.0475m and it is surrounded by a coppery solid coil. The test case is implemented on a work station with Intel Xeon E5-2698 v4 CPU and NVIDIA Tesla V100-PCIR-16GB GPU. Material properties and voltage to generate excitation current are given in Table 3.1.

For the given problem definition, computation is carried out based on the following parameters. The relative tolerance for TLM iteration is set to 10^{-5}. Time-step for the case study is set to 1/1200s and time length is 0.25s.

After post-processing, the magnetic flux density vector field is obtained. The field dis-

Table 3.1: Problem definition

Material information		
Domain	σ	B-H curve
Air	0	$H = \upsilon_0 B$
Coil	10^7S/m	$H = \upsilon_0 B$
Core	10^4S/m	H= $\begin{cases} \upsilon_0 B/2000 \ \ \lvert B\rvert < 1.3 \\ \upsilon_0[\lvert B\rvert + (\lvert B\rvert - 1.3)^8] \times \lvert B\rvert/2000 B \lvert B\rvert \geq 1.3 \end{cases}$
v(t)	$0.45sin(120\pi t)V$	

tribution is displayed in Fig. 3.5 at the time when the maximum value occurs.

A comparison is made with $Comsol^{TM}$ to verify the precision and efficiency of the proposed algorithm and the result shows good accuracy with average relative error less than 2% over all space and time span. Fig. 3.6 and Fig. 3.7 displays the field results obtained from both $Comsol^{TM}$ and the TLM-FEM scheme.

Meanwhile, a significant speed-up can be seen for different mesh sizes (Table 3.2). It is not surprising to witness an excellent speed-up since the proposed 3D-TLM-FEM scheme, in nature, has excellent parallelism and the V100 GPU has over 5000 cores. However, it is still worth mentioning that the speed-up depends on TLM iterations needed per time-step and may vary for different B-H curves and excitation amplitudes. Also, different matrix solution algorithm affects the solution time.

Table 3.2: Execution Time and Speedup of 3D-TLM

Mesh	Number of			Execution time		Speed
ID	Elements	DOFs	Nonlinear DOFs	$Comsol^{TM}$	TLM	up
1	7870	9061	1594	191.1s	3.65s	52.1
2	12533	17423	2604	401.9s	9.69s	41.5
3	20249	27996	4165	736s	24.7s	29.8
4	39258	54559	6109	2636s	162.1s	16.3
5	66706	91664	9236	4750s	342.8s	13.9
6	143964	196973	14283	11464s	912.7s	12.6

3.7 Discussion

As is seen in Table 3.2, speed-up decreases as mesh complexity increases. This leads to the following questions and discussion: Is the performance hindered by TLM algorithm itself for the case study? Will the 3D-TLM-FEM method remain efficient for larger mesh size? More analysis/discussion is given below.

For time analysis, the following approximation is used:

$$T_{TLM} \approx N_{TLM} \times (t_{NR} + t_{LU}), \tag{3.29}$$

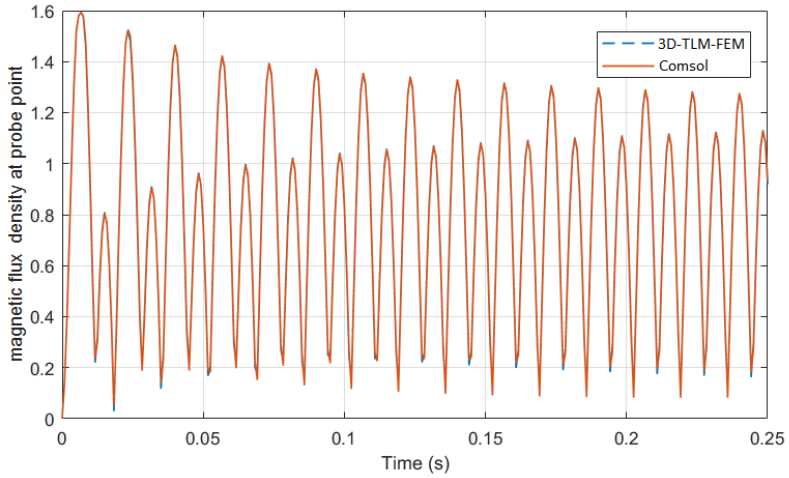


Figure 3.6: Comparison of magnetic flux density module at probe point vs. time.

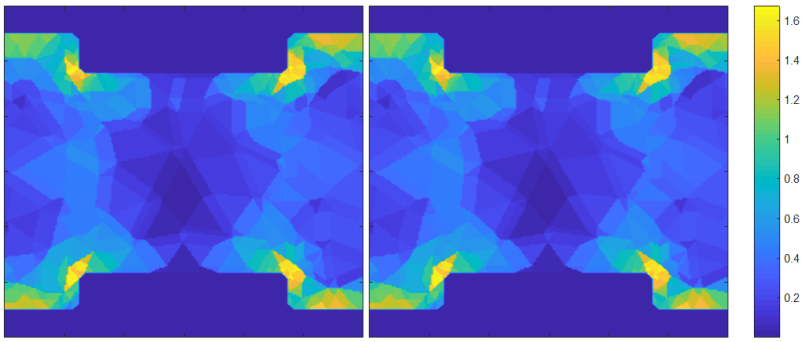


Figure 3.7: Comparison of magnetic flux density module at Z=0.33 and t=0.00667 (left from the proposed 3D-TLM-FEM scheme and right from $Comsol^{TM}$).

where T_{TLM} is the total TLM execution time, N_{TLM} is the total TLM iterations needed for entire computation, t_{NR} is the NR iteration time at one single iteration and t_{LU} is the LU triangle matrix back substitution time. t_{NR} and t_{LU} stays almost unvaried because LU matrix/routine and scale of NR iteration stays the same during the entire TLM iteration.

To find the reason for the speed-up drop, N_{TLM} for the case study is investigated. By comparing information in Fig. 3.8 and Table 3.2, it is obvious that the increase of T_{TLM} vs. DOF is much higher than that of N_{TLM}. According to (3.29), a nonlinear growth of $t_{NR} + t_{LU}$ is seen, and the timeline generated by $CudaVisualProfiler^{TM}$ indicates that t_{NR} stays almost the same for all meshes. Therefore, the nonlinear increment of t_{LU} hinders the computation for a larger mesh, possibly due to the insufficient number of cores to fully parallelize triangle matrix back substitution. It is the LU solving process, rather than TLM-FEM scheme, that causes the majority of speed reduction for the case study. With a better algorithm and more powerful GPU, the LU triangular solution time is likely to approach ideal $O(DOF)$ and the performance of the case study will increase dramatically.

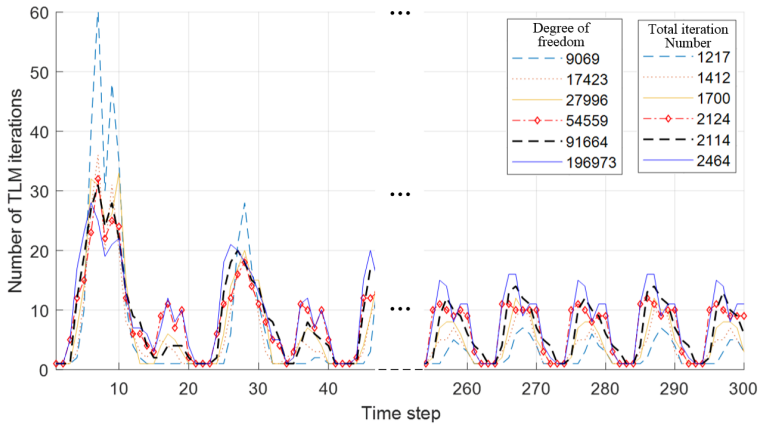


Figure 3.8: TLM iteration number at each time-step for the case study.

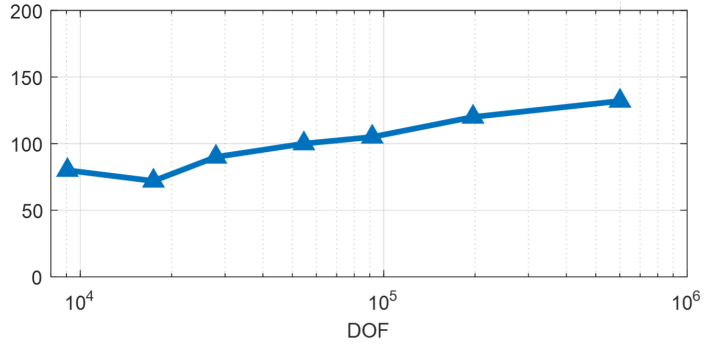


Figure 3.9: TLM iterations required to reach tolerance of 10^{-5} in static scenario.

N_{TLM} is not the 'black sheep' in the case study, but is it possible that N_{TLM} grows impractically large for bigger problems? To eliminate the interference of time-stepping, another comparison of N_{TLM} between different mesh sizes is carried out for a static case where excitation current drives the inductor into deeper saturation, and results in Fig. 3.9 show that TLM iteration number is not sensitive to mesh DOF. Even an initial descending trend of N_{TLM} vs. DOF is witnessed. This coincides with the intuition that TLM iteration is much more related to matching than network complexity.

Based on the above discussion, the proposed 3D-TLM-FEM scheme is promising, especially for large mesh since its algorithm complexity comes close to $O(DOF)$. Furthermore, for long-time-span transient simulations, the 3D-TLM-FEM will gain more advantage — N_{TLM} reduces at later time-steps and the difference between N_{TLM} narrows for different mesh sizes, as is shown in Fig. 3.8. The increment of TLM iterations is much smaller than that of mesh DOF. With a good convergence rate and an iterative nature, the 3D-FEM-TLM scheme may look similar to the quasi-Newton method, however, they are essentially 2 different methods.

3.8 Summary

In this chapter, the transmission line method was successfully extended for the 3D nonlinear electromagnetic field analysis using edge vector finite elements. The challenges were discussed and conquered, and it turned out that the good features of the TLM such as constant admittance matrix and massive parallelism can also benefit the computation of 3D nonlinear edge vector elements in practice.

More specifically, the transmission line modeling was applied to the edge-element-discretized formulation to calculate the EM field with nonlinearity. To solve the discretized formulation with excellent parallelism, nonlinearity and matrix rank were properly handled. The concept of scattering box was introduced to model elemental local nonlinearity for the TL-circuit system. Transmission lines in the system successfully decoupled local nonlinearity from the global linear network so that computation can be parallelized at the 3D element level. In addition, the proper gauge was applied to edge finite element formulation, which results in a full ranked matrix system to allow LU factorization of the global linear network, making it possible to carry out matrix factorization only once during the whole computation process.

Due to the above property, the proposed 3D-TLM-FEM is perfectly suitable for high-performance parallel computation, and a comparison between the TLM implementation and $Comsol^{TM}$ shows excellent speed up (over 50) while maintaining a good precision for the power inductor case study.

4

Domain Decomposition of 3-D Edge Elements

4.1 Introduction

This chapter explores the edge domain decomposition method to parallelize the finite element computation at the edge level. The background, prerequisite knowledge, and derivation of the method are given. The implementation of the edge domain decomposed FEM shows excellent suitability for SIMD architectures.

The finite element method (FEM) has drawn increasing attention in power system simulations due to its superior precision and ability to handle complex geometries. However, the large-scale nonlinear FEM matrix system usually requires a significant amount of computational power during global Newton Rapson (NR) iterations, as is illustrated in chapter 1. Moreover, if the external circuit equations are integrated into the global NR Jacobian matrix, the solvability of the matrix system can be easily damaged [33].

The development of many-core high-performance computation hardware provides possibilities to improve the efficiency of FEM models in power system simulations, and many different methods were proposed to adapt the time-consuming nonlinear FE algorithm for parallel architectures. For example, the domain decomposition methods, including the overlapping Schwartz scheme [34] and the non-overlapping FETI [35] scheme, divide the global system into several smaller sub-domain problems so that the lighter sub-domain matrices can be solved in parallel. To accelerate the solution of those matrices, the super LU / paralleled Conjugate Gradient method was introduced. To avoid frequent updating for nonlinearity in those matrices, the transmission line modeling method [36] was implemented.

However, the above methods still need large matrices and massive parallelism is hardly achieved, since piecewise information, generated from each domain element, is integrated

into one big system and is solved simultaneously in a single step. This is a centralized way of thinking. A decentralized thinking pattern, in contrast, could fully unlock the computational power of massively-parallelized architectures. For example, the GPU-based algorithm achieved element-wise parallelism to accelerate matrix-vector multiplication [37], and a decent speed-up was observed. In addition, a novel decentralized scheme called nodal domain decomposition relaxation (NDDR) was introduced recently for 2-D nodal triangular FEM [16]. The method has the following key features. All sub-domains are shrunk into a minimum size at the single-node level, and each sub-domain has only 1 degree of freedom that can be independently solved. Also, each sub-domain only communicates with its neighboring domains in a distributed manner. Benefitting from the above properties, the method allows the handling of material nonlinearity during iterations, as well as nodal-level parallelism and matrix-free computation.

Despite the excellent modularity for massively parallelized architectures, the above nonlinear NDDR method was only developed for 2-D triangular elements. While the 2-D nodal element and geometry lead to significant error and deviation from reality, as is explained in section 1.5. Thus, it is important to explore new algorithms to integrate the above decentralized domain decomposition scheme with 3-D edges elements, especially for nonlinear transient field-circuit co-simulation where a huge amount of computational resources is needed.

In this chapter, we extend our former effort of the 2-D NDDR scheme [16] further to the nonlinear 3-D edge elements. It shares the similar decentralized idea of applying domain-decomposition to each node and element. However, rather than solving a simple 2-D problem with scalar nodal unknowns, we achieved edge-level parallelism on the reduced magnetic vector potential (RMVP) for 3-D vector elements with much larger degree of freedoms. Different convergence and gauging behaviors are also explored. Furthermore, We purpose a field-circuit coupling scheme which allows large eddy current and separate solution of circuit and FE systems with high accuracy. Inductor and transformer cases are studied to verify the accuracy and efficiency of the proposed EDD and field-circuit coupling scheme, and comparison with $Comsol^{TM}$ indicates a good accuracy and decent speed-up of over 43.

The chapter is arranged as follows. Section 4.2 introduces the eddy current RMVP formulation discretized by edge element interpolation functions. In 4.3, the decentralized idea is extracted from traditional schemes and is extended as the new EDD method. Section 4.4 proposes the field-circuit coupling technique, which is then integrated with the EDD method in the case studies shown in Section 4.5. Finally, Section 4.6 and 4.7 give the discussion and the summary of this chapter.

4.2 FEM Formulation For Eddy Current Analysis

4.2.1 Reduced Magnetic Potential Formulation

As is introduced in chapter 2 and chapter 3, instead of directly dealing with the quasi-static Maxwell's equations, one can solve the following for low-frequency magnetic problems:

$$\nabla \times (\upsilon \nabla \times \vec{A}) + \sigma \frac{\partial \vec{A}}{\partial t} - \vec{J}_e = 0, \tag{4.1}$$

where υ is the field-dependent reluctivity, σ is the electrical conductivity, and $\vec{J}_e$ is the impressed current density.

4.2.2 Finite Elements And Discretized Formulation

When the finite element method is applied, the above equation can be simplified into a limited degree of freedoms (DOFs). To achieve this, the edge domain decomposition scheme follows a similar derivation process presented in chapter 2. However, for concise notation, the element number index e in chapter 2 is omitted ($A_i^{(e)}$ or A_{e_i} changes to A_i, $K_{e_i e_j}$ changes to K_{ij}), and the 6×6 matrix-form elemental discretized formulation as follows:

$$[K_{ij}]_{6\times6}[A_j]_{6\times1} + [D_{ij}]_{6\times6}\frac{\partial}{\partial t}[A_j]_{6\times1} = I[f_i]_{6\times1} + [\tau_i]_{6\times1}, \tag{4.2}$$

where

$$K_{ij} = \int_{Ve} \nu^e \left(\nabla \times \overrightarrow{N_i}\right) \cdot \left(\nabla \times \overrightarrow{N_j}\right) dV, i, j \in [1, 6], \tag{4.3}$$

$$D_{ij} = \int_{Ve} \sigma^e \overrightarrow{N_i} \cdot \overrightarrow{N_j}\, dV, i, j \in [1, 6], \tag{4.4}$$

$$f_i = \int_{Ve} \overrightarrow{N_i} \cdot \overrightarrow{J_{unit}}\, dV, i \in [1, 6], \tag{4.5}$$

$$\tau_i = \int_{Se} \left(\nu \nabla \times \vec{A}\right) \times \overrightarrow{ds} \cdot \overrightarrow{N_i}, i \in [1, 6], \tag{4.6}$$

and V_e is the volume, S_e is the surface boundary of the element, $\overrightarrow{J_{unit}}$ is the coil current density at 1 Amp coil current, and I is known current in the coil which changes with time.

The time derivative in (4.2) can also be discretized using the Backward Euler method:

$$[M_{ij}]_{6\times6} \cdot \left[{A_j}^t\right]_{6\times1} = [b_i]_{6\times1} + [\tau_i]_{6\times1}, \tag{4.7}$$

where

$$[b_i]_{6\times1} = I[f_i]_{6\times1} + [D_{ij}]_{6\times6}/\Delta t \cdot \left[{A_j}^{t-1}\right]_{6\times1}, \tag{4.8}$$

$$[M_{ij}]_{6\times6} = [K_{ij}]_{6\times6} + [D_{ij}]_{6\times6}/\Delta t, \tag{4.9}$$

and Δt is the time-step length, ${A_j}^t$ is the unknowns to solve for at current time-step, ${A_j}^{t-1}$ is A_j at the previous time-step.

When the element is inside a ferromagnetic material region, the ν_e become dependent on magnetic flux density $\vec{B}$, and $\vec{B}$ is a function of A_i. This means that (4.7) becomes a nonlinear equation of A_i. The common NR iteration can be used to solve the nonlinear system. According to the method, (4.7) becomes a form of Jacobian matrix and residual. The following equations are necessary to form the Jacobian matrix:

$$\vec{B} = \nabla \times \vec{A} = \sum_{i=1}^{6} A_i \cdot \nabla \times \vec{N_i} = \sum_{i=1}^{6} A_i \cdot \frac{2\overrightarrow{\nabla L_{i1}} \times \overrightarrow{\nabla L_{i2}}}{l_i}, \tag{4.10}$$

$$\frac{\partial \nu_e}{\partial A_i} = \frac{\partial \nu_e}{\partial B^2} \cdot \frac{\partial B^2}{\partial A_i}, \tag{4.11}$$

where $\frac{\partial \nu_e}{\partial B^2}$ is determined by B-H curve of the material.

However, the above discussion only gives the restriction of 6 local A_i inside one single element. To achieve solution over the entire domain, all elements must communicate with each other. The communication is subject to Maxwell's equations:

1. Every edge has only 1 global direction and value A_k. When one edge is shared by multiple elements, all local A_is of such edge in different elements are equal to A_k. And all local $\vec{N_i}$s adjust signs so that their components on the edge all point in the global direction. This ensures the continuity of tangential $\vec{A}$, thus normal $\vec{B}$ is continuous between elements [19].

2. On a triangular surface shared by 2 elements, the τ_is in the 2 elements are opposite to each other. As a result, tangential $\nu\nabla \times \vec{A}$ or $\vec{H}$ is continuous between elements.

How to effectively handle the above elemental nonlinear equation and inter-element communication consistency presents a challenge to different methods. This will be discussed in the next part.

4.3 Edge Domain Decomposition To Solve The Global Nonlinear System

As is shown in Fig. 4.1(a), the traditional method only has one domain for the whole solution area. The method explicitly enforces elemental restrictions and global consistency through the assembly process. During assembly, one global matrix system is formed, where each edge only has one global unknown (the global counterpart of multiple elemental unknowns sharing the edge). Elemental matrices and vector are added directly to the global matrices/vector at the position of the global counterparts, and the surface integration of (4.6) does not appear in the system since they are canceled out. Thus all local and inter-element restrictions are respected. After the application of a proper global domain boundary condition, the matrix system can be solved.

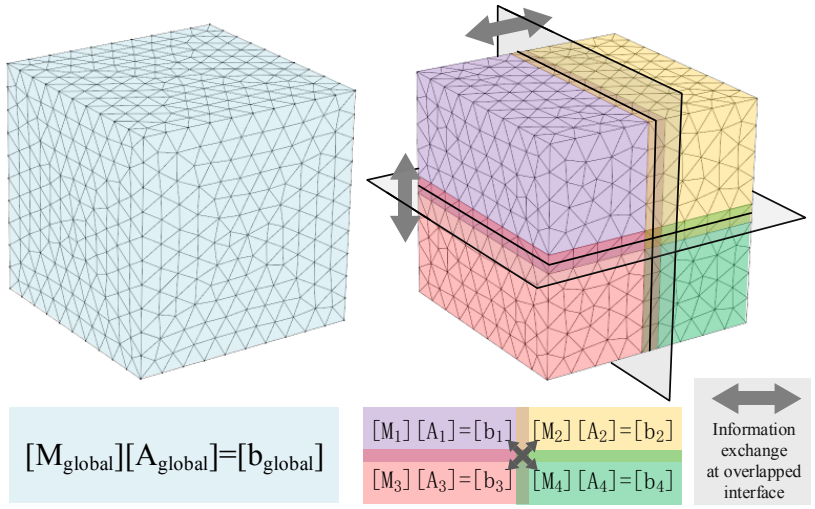


Figure 4.1: (a) Left side: traditional single domain FEM. (b) Right side: overlapping domain decomposition method with 4 sub-domains marked in different colors.

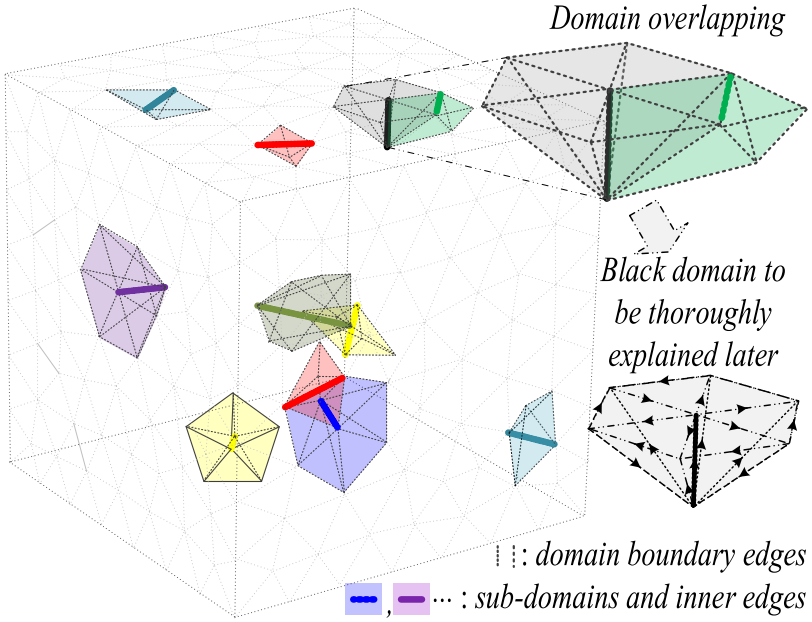


Figure 4.2: EDD scheme with sub-domains containing only one unknown edge element.

4.4 Edge Domain Decomposition To Solve The Global Nonlinear System

However, the sparse global matrix may have several millions of unknowns (total number of edges in the domain). In addition, the matrix system must be globally assembled and solved at every NR iteration, which could be computationally expensive even for efficient matrix solvers.

Such costs can be reduced using a traditional domain decomposition method. As is

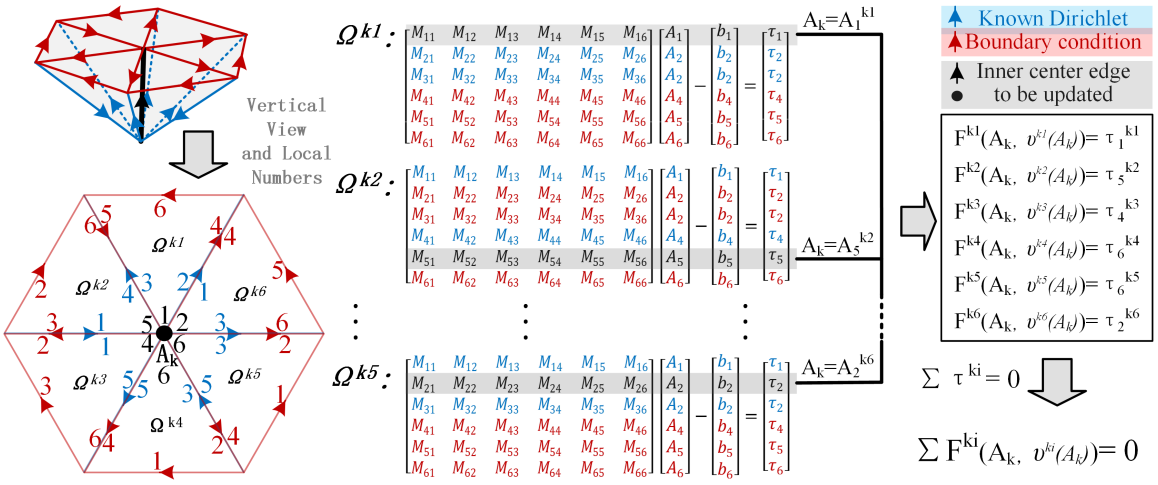


Figure 4.3: Detailed formulation within each sub-domain for the EDD scheme.

shown in Fig. 2(b), the whole solution region is divided into several sub-domains. And similar assembly happens in each sub-domain to generate smaller matrix systems. These sub-domain systems can be solved in parallel and different schemes [35], [34] may be applied to exchange information and reach consistency (inter-element consistency) on sub-domain interfaces.

For example, if Schwartz scheme [34] is applied, the parallelism and consistency are ensured by an overlapping/iteration technique. Under this scheme, the interface is a thin layer of elements, which means any given sub-domain's interface boundary locates inside other sub-domains. All sub-domain's interface boundary is given a guessed initial value at the start of the iteration. With global/interface boundary conditions, each sub-domain system is then solved independently to generate a domestic result on its inner edges, which is used to update other sub-domain's Dirichlet boundary condition at the next iteration step. Such iteration repeats until a consistent global result is reached.

Due to reduced problem size and inter-sub-domain parallelism, the DD method has significantly improved efficiency for some parallel computing architectures (such as multi-core CPU). However, the sub-domain systems, depending on the domain partition, can still possibly have thousands of unknowns [35]. And a large nonlinear sub-domain system may still require expensive and repetitive assembly and solving at NR iterations. In addition, extra computation burden may be caused by special techniques to handle inter-sub-domain consistency. These properties make the DD implementation not suitable on GPU architecture, which is designed for massively parallelism of light-weight tasks.

To expand parallelism and reduce single-core workload, a natural idea is to shrink the size of each sub-domain. Fig. 4.2 shows an extreme situation where each edge has its own sub-domain and each domain only consists of elements sharing the edge. In each domain, there is only 1 internal edge. This means the sub-domain system becomes a super light 1×1 equation after the Dirichlet boundary condition is applied. When one applies the

above overlapping/iteration scheme, those Dirichlet boundaries are simply the respective neighbor edges' values at the previous iteration step. Due to the overlapping, each edge is updated based on its direct neighbors and, meanwhile, serves as boundary conditions when its neighbors are updated at the next iteration step (see the black and green edge in Fig. 4.2). A consistent global result can be reached after iterations. Therefore, each internal edge can be updated independently by a 1×1 equation, and edge level parallelism is achieved.

Fig. 4.3 explains how the 1×1 equation is formed for the black example sub-domain. The Hexagonal pyramid domain includes 6 elements and 19 edges. Each edge has its global direction labeled in arrows (upper red and lower blue). For easier interpretation, the 3D shape is projected into a plane with elemental local edge numbers displayed. The only domain inner edge becomes the center black dot. Each elemental system only contributes its inner-edge row into the 1×1 equation. For example, the inner edge (A_k) is numbered 1 inside element Ω^{k1}. Thus, in matrix Ω^{k1}, only the first row is valid because the other 5 edges' rows are eliminated as boundary conditions. The other 5 elements follow similar pattern. Since tangential $\vec{H}$ continues on all surfaces sharing edge A_k, sum of black τ_is becomes 0. This gives rise to the following equation:

$$\sum\nolimits_{i=1}^{N} F^{ki}\left(A_k\right) = 0, \tag{4.12}$$

where k is the global index of the edge to be solved, N is the total number of neighboring elements sharing edge k , ki is the element index of its i^{th} neighboring element, and F_{ki} is the inner-edge row of the i^{th} neighboring element. Note that the equation only has one unknown A_k, and during its assembly, inter-element consistency is well respected by explicitly applying the restrictions. When NR method is used to solve the nonlinear (4.12), the increment of A_k is calculated by:

$$\Delta A_k = \frac{\sum_{i=1}^{N} F^{ki}\left(A_k\right)}{\sum_{i=1}^{N} \frac{\partial F^{ki}(A_k)}{\partial A_k}} \tag{4.13}$$

Based on the above discussion, the EDD scheme handles nonlinearity, inter-element consistency, while simultaneously allowing massive parallelism and a light-weight single-core task. Moreover, no global matrix is needed because all information needed for the single-edge calculation can be drawn from its neighbors. The calculation program flowchart is shown in Fig. 4.5. To effectively access data for the program, the C language structures are defined in Fig. 4.4. Each element and edge consumes a fixed memory space. As a result, the total memory needed is linear to the problem size.

It is also worth mentioning that the proposed 3D-EDD scheme is equivalent to Jacobi iteration under linear cases, which is similar to the N-scheme in [38]. However, the 3D-EDD can easily integrate nonlinearity and matrix solution process without significantly affecting the convergence rate. Also, since the EDD method is originated from domain

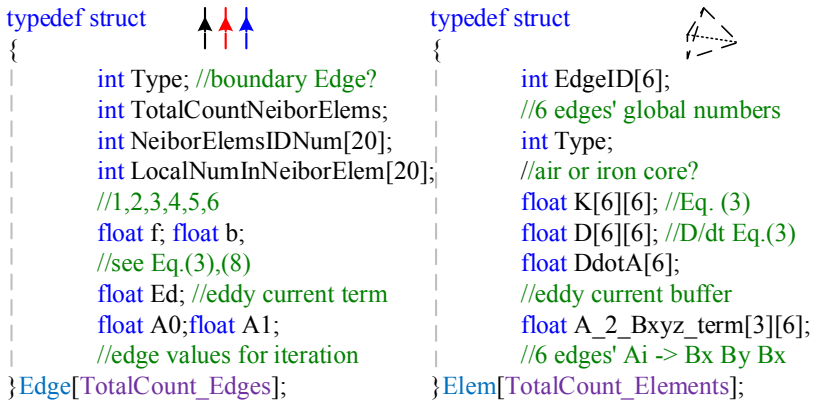


Figure 4.4: Data arrangement of EDD scheme in C language.

decomposition, replacement or modification of the sub-domain boundary condition may dramatically increase the convergence speed.

4.5 Coupling Scheme For Field-Circuit Co-Simulation

The above edge domain decomposition scheme can provide electromagnetic field distribution based on an input coil current amplitude. However, coil currents in a power device (such as transformers) are always from a power system circuit, and a proper field-circuit coupling scheme is still necessary to interface with the external drive circuit.

There are 2 types of coupling schemes [33]: direct and indirect methods. The direct method simultaneously solves the entire FE and circuit system in one global matrix, which is intuitive and precise, and a symmetric matrix may be generated [39]. However, the method requires FE matrix and it may destroy the iteration convergence of the global matrix, which is not suitable for the matrix-free iterative EDD scheme. On the other hand, indirect methods allow separate solutions of the circuit and FE systems. But some of the previous work cannot precisely handle strong eddy currents [17].

In this work, we propose an indirect coupling scheme by observing the time-discretized restrictions on coil current and voltage through traditional FE matrices. The scheme solves field and circuit equations separately and can produce accurate results under strong eddy currents.

The voltage of a 3D coil results from the electric field at the direction of coil wire, which is also equivalent to time derivative of coil Magnetic flux φ:

$$V = \frac{N_{coil}}{S_{coil}} \cdot \int_{V\,coil} \frac{\partial \vec{A}}{\partial t} \cdot \overrightarrow{n_{coil}}\, dv = \frac{\partial \varphi}{\partial t}, \tag{4.14}$$

where N_{coil} is the number of coil windings, S_{coil} is the wire intersection area, $\overrightarrow{n_{coil}}$ is the unit vector of predefined coil wire direction. When the $\vec{A}$ field is discretized by known

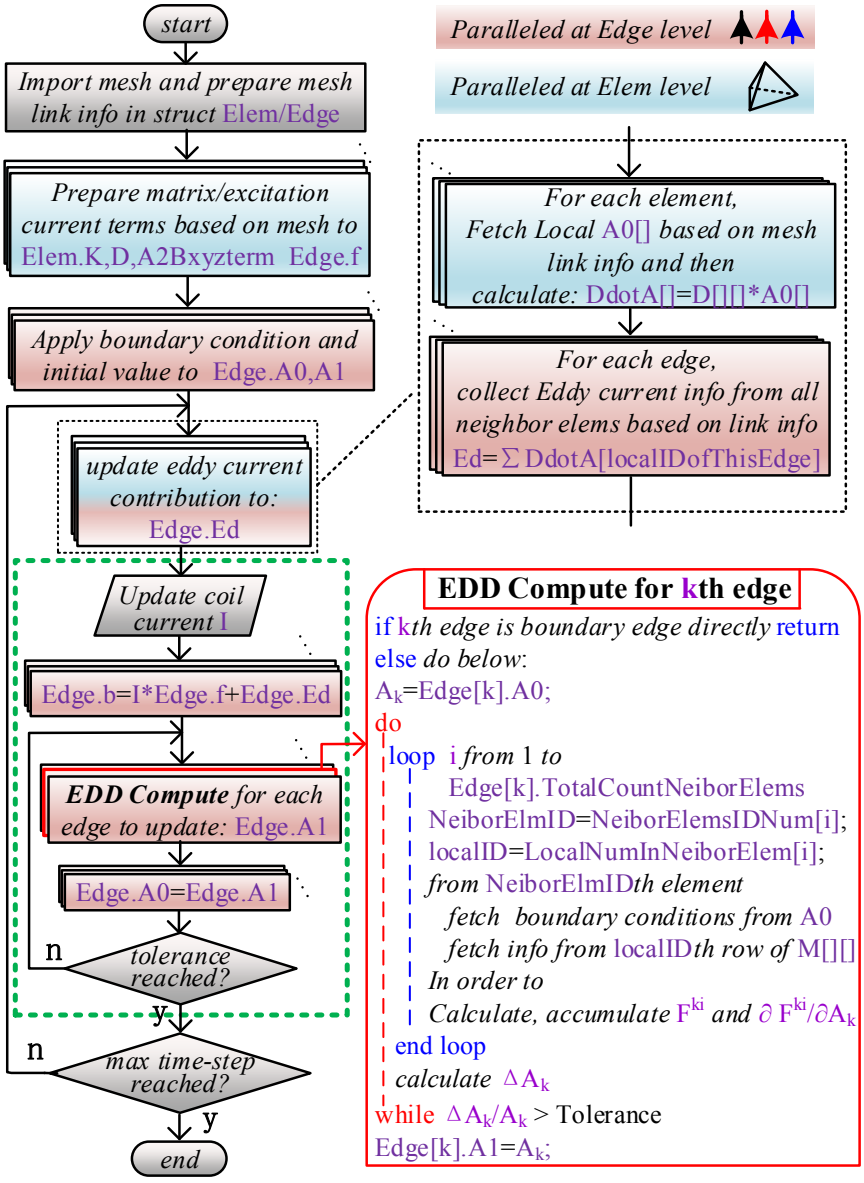


Figure 4.5: Flowchart of EDD scheme.

interpolation functions, the above equation becomes

$$V = \sum_{k=1}^{Q} \frac{\partial \mathrm{A_k}}{\partial t} \left(\int_{V coil} \frac{N_{coil}}{S_{coil}} \cdot \overrightarrow{N_k} \cdot \overrightarrow{n_{coil}} \, dv \right), \tag{4.15}$$

where Q is the total number of edges, A_k is the field unknown on k^{th} global edge, $\overrightarrow{N_k}$ is interpolation functions associated with k^{th} edge. As a result, the voltage becomes a linear combination of unknowns on each edge noted as:

$$V = \mathbf{A}\_\varphi \cdot \frac{\partial \mathbf{A}}{\partial t}, \tag{4.16}$$

where $\mathbf{A}$ is a column vector of all edge unknowns, and $\mathbf{A}\_\varphi$ is the row vector of the integration terms in (4.15), which can be found before FEM solution. Note that $\mathbf{A}\_\varphi$ also maps

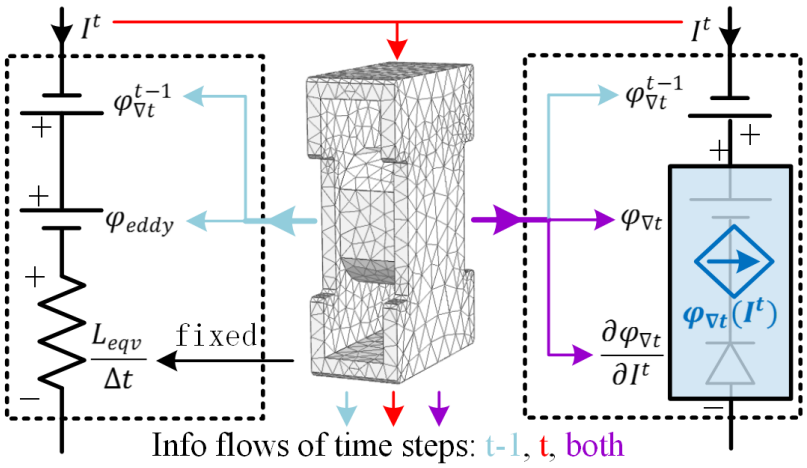


Figure 4.6: FE circuit models for circuit coupling: left side is for linear cases, and right side is for nonlinear ones.

A into coil flux φ.

The (4.16) and the FEM global matrix system establish the link between coil current and voltage, with **A** as bridge. This relation is used to abstract the FEM model into a circuit component by several steps:

$$\begin{bmatrix} \mathbf{K} & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} A \\ V \end{bmatrix} + \begin{bmatrix} \mathbf{D} & 0 \\ \mathbf{A}\_\varphi & 0 \end{bmatrix} \cdot \begin{bmatrix} \partial \mathbf{A}/\partial t \\ V \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} I(t), \tag{4.17}$$

where $\mathbf{K}$ is the $N \times N$ global stiffness matrix, $\mathbf{f}$ is the $N \times 1$ global excitation column vector of the coil, assembled by elemental equations (4.2). When Backward Euler method is used to discretize the time derivative, the above equation becomes:

$$\begin{bmatrix} \mathbf{K} + \frac{\mathbf{D}}{\Delta t} & 0 \\ \frac{\mathbf{A}\_\varphi}{\Delta t} & -1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A}^t \\ V^t \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix} I^t + \begin{bmatrix} \frac{D}{\Delta t} & 0 \\ \frac{\mathbf{A}\_\varphi}{\Delta t} & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A}^{t-1} \\ V^{\,t-1} \end{bmatrix}, \tag{4.18}$$

where the upper index t means unknowns of current time-step, and t-1 means knowns from previous the time-step. After basic linear algebra operations, one can obtain:

$$V^t = \psi \left(\mathbf{f} \times I^t + \mathbf{D}/\Delta t \cdot \mathbf{A}^{t-1} \right) - \varphi_{dt}^{t-1}, \tag{4.19}$$

where $\varphi_{dt}^{t-1} = \mathbf{A}\_\varphi/\Delta t \cdot \mathbf{A}^{t-1}$, and the row vector $\psi = [\mathbf{A}\_\varphi/\Delta t] \cdot [\mathbf{K} + \mathbf{D}/\Delta t]^{-1}$. Note that multiplying the row vector ψ with a column vector can be interpreted in another way: solve the FEM problem with the column vector as excitation, and extract the coil flux term from the solution vector $\mathbf{A}^t$. This process is defined as ψ operation to a column vector. The equation explicitly describes the link between V^t and I^t, which directly give rise to the following circuit models.

For linear cases, operator ψ is fixed. Therefore, (4.19) degenerates into a pure linear restriction:

$$V^t = L_{eqv}/\Delta t \times I^t + \varphi_{eddy} - {\varphi_{\Delta t}}^{t-1}, \tag{4.20}$$

where $L_{eqv} = \psi \cdot \mathbf{f}$, $\varphi_{eddy} = \psi \left(\frac{\mathbf{D}}{\Delta t} \cdot {\mathbf{A_k}}^{t-1}\right)$. The FEM system becomes a fixed linear resistor and voltage sources (extracted from history FEM solution) in the circuit, as is shown in Fig. 4.6 left. For nonlinear cases, operator ψ changes with its input. However, since $\mathbf{A}^{t-1}$ is known from previous time-step, ψ becomes a nonlinear function ($\varphi_{\Delta t}$) of I^t, and (4.19) degenerates into:

$$V^t = \varphi_{\Delta t}\left(I^t\right) - {\varphi_{\Delta t}}^{t-1}. \tag{4.21}$$

Thus the FE system is regarded as a nonlinear current-controlled voltage source and a fixed voltage bias in the circuit shown in Fig. 4.6 right.

With the above FE circuit model, the FE and circuit systems can be solved separately. From the circuit solver's perspective, the complex FEM solution process is abstracted away as different circuit components and the FE information comes back to the circuit as voltages on those components. On the other hand, the FE solving process only sees current I^t from the circuit as input (operator ψ only sees I^t).

However, despite the convenience of isolation, the model still needs iterations (usually 2-4 times) to handle the nonlinearity. The purpose of the iterations is to find an I^t that leads to consistent voltages on both the FE circuit model and the circuit connected to it. If the NR scheme is applied, the circuit solver needs to know $\frac{\partial \varphi_{\Delta t}\left(I^t\right)}{\partial I^t}$ and $\varphi_{\Delta t}\left(I^t\right)$ to calculate the increment of I^t. The $\varphi_{\Delta t}\left(I^t\right)$ can be obtained by directly solving the FEM problem at coil current I^t, and there are different methods to find $\frac{\partial \varphi_{\Delta t}\left(I^t\right)}{\partial I^t}$. For simplicity, the small probing increment method shown in (4.22) is used in this work:

$$\frac{\partial \varphi_{\Delta t}\left(I^t\right)}{\partial I^t} = \frac{\varphi_{\Delta t}\left(I^t + dI^t\right) - \varphi_{\Delta t}\left(I^t\right)}{dI^t}. \tag{4.22}$$

Note that the calculation of $\varphi_{\Delta t}\left(I^t + dI^t\right)$ and $\varphi_{\Delta t}\left(I^t\right)$ can be parallelized due to independence, and the iteration flow chart is shown in Fig. 4.7.

The iteration converges to the same result of the direct coupling method, regardless of eddy current strength. The reason is that the proposed scheme simultaneously enforces the field, circuit, as well as the voltage/current consistency equations, and the same restrictions are explicitly assembled into the global matrix in the direct coupling methods. Also, note that although the scheme derives from matrices, it is still valid for matrix-free EDD FEM due to independent solutions of field and circuit.

4.6 Case Studies

To demonstrate the efficiency and precision of the above methods, two case studies were carried out and results were compared with $Comsol^{TM}$ on the same mesh. The first one verified the nonlinear handling ability of the EDD scheme in a deep-saturated static scenario, and the second one combined field-circuit coupling and EDD in a transient simulation. The algorithm was developed in CUDA-C language, and the test was imple-

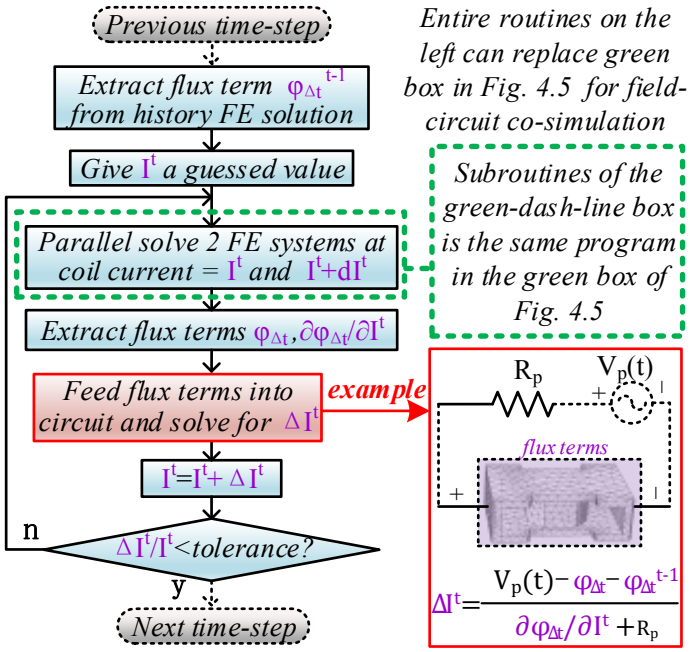


Figure 4.7: Field-circuit coupling iteration flow chart at time-step t.

mented on Intel Xeon E5-2698 v4 CPU ($Comsol^{TM}$) and NVIDIA Tesla V100-PCIR-16GB GPU (EDD scheme).

For the computation parameters, the termination condition is set to a global relative change of 1e-6 and a global relative residual of 1e-5, and all domains have Dirichlet boundary condition for the tangential component of $\vec{A}$:

$$\overrightarrow{A_{\parallel}} = 0. \tag{4.23}$$

4.6.1 Nonlinear Static EDD Simulation

Table 4.1: Inductor Problem definition

Coil/air permeability	μ_0	Coil current/winding turns	16A/380
Iron core B-H curve	H=	$\begin{cases} v_0 B/2e3, \ \|B\| \le 0.6 \\ v_0 B/2e3 + 4e4(\|B\| - 0.6)^4 \|B\|/B, \|B\| > 0.6 \end{cases}$	

Fig. 3.4 shows a power inductor with a blue iron core and a copper coil. Since it's studied in a static case, the D matrix in (3) was set to 0. The material and coil parameters are given in Table 4.1.

After computation and post-processing, the field is displayed in Fig. 4.8, and the comparison with $Comsol^{TM}$ over different mesh size is shown in Table 4.2. The result shows good accuracy with an average relative error of less than 2% over space domain. Meanwhile, a significant speed-up can be seen due to excellent parallelism and a fast conver-

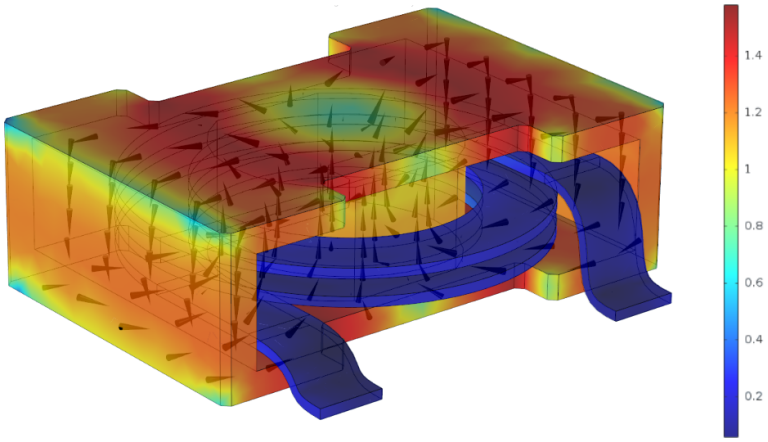


Figure 4.8: Saturated static magnetic flux density (T) of the power inductor.

gence rate over DOFs. However, readers should notice that the speed-up may vary for different B–H curves and excitation amplitudes, since the convergence rate will be affected by the system spectral radius.

It is also worth mentioning that the static version of (4.2) does not have a full rank because it cannot specify the divergence of $\vec{A}$. However, the static case study still converges to the correct solution. This means that the iteration process can auto-gauge the problem like the conjugate gradient method [40].

Table 4.2: Comparison With $COMSOL^{TM}$ For The Static Case

Mesh size (DoF)		7870	24.3k	47.5k	79.8k	172k	523k
Time use(s)	$Comsol^{TM}$	14.9	47.2	107.5	198.1	579.3	2274
	EDD	0.34	1.26	3.82	11.9	38.7	206.9
EDD iteration		5158	5529	7019	9039	13.8k	24.7k
Speed-up		43.7	37.5	28.2	16.6	14.9	10.99

4.6.2 Nonlinear Dynamic Field-Circuit Co-Simulation

As is shown in Fig. 4.9, the studied transformer has 3-phase metal-colored coils and an E-shape blue iron core. The transformer has 6 coils while the above coupling scheme only describes the voltage-current relation of a single coil, which means further extension should be made. Six coils, rather than a single one, all contribute to the total external excitation current in the finite element domain. The influence of 6 coil currents are packed into a total column FE excitation vector:

$$\mathbf{f_{all}} = \sum_{i=1}^{6} {I_i}^t \times \mathbf{f}_i, \tag{4.24}$$

where ${I_i}^t$ is the current inside i^{th} coil, and $\mathbf{f_i}$ is the FE global excitation vector of i^{th} coil. $\mathbf{f_i}$ is then fed into the FE solver to obtain the field distribution ($\mathbf{A}$), which generates voltages

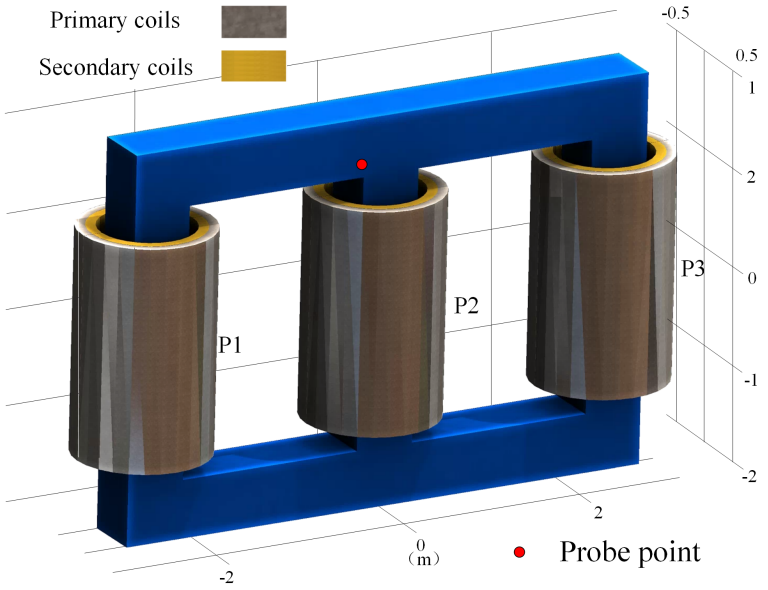
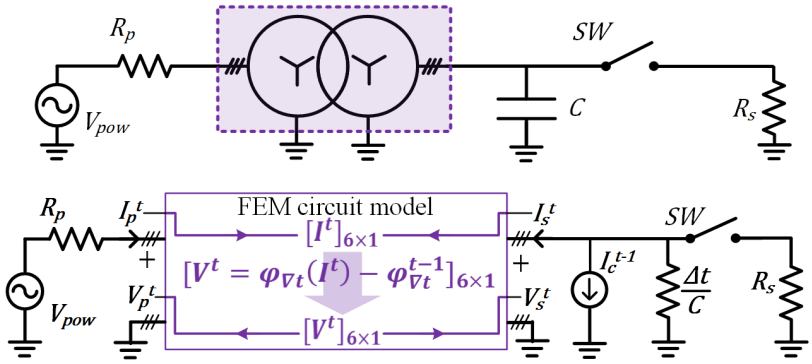


Figure 4.9: Geometry of the three-phase transformer in meters.



$$\begin{bmatrix} \boldsymbol{R_p} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R_{eq}} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\Delta I_p}^t \\ \boldsymbol{\Delta I_s}^t \end{bmatrix} + \begin{bmatrix} \frac{\partial \varphi_{\Delta t}}{\partial I^t} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\Delta I_p}^t \\ \boldsymbol{\Delta I_s}^t \end{bmatrix} = \begin{bmatrix} \boldsymbol{V_{pow}} \\ \boldsymbol{R_{eq}} \cdot \boldsymbol{I_c}^{t-1} \end{bmatrix} - [\boldsymbol{V^t}]$$

, where $R_{eq} = R_s \parallel \frac{\Delta t}{C}$, all $\boldsymbol{R}$ are 3×3 diagonal matrices, $\boldsymbol{I}$ and $\boldsymbol{V}$ are 3×1 vectors of the respective 3-phase quantity, $\frac{\partial \varphi_{\Delta t}}{\partial I^t}$ is the 6×6 Jacobian matrix of the FE model.

Figure 4.10: Test circuit with FE model and the equation to calculate current increment.

on all 6 coils:

$$\mathrm{V}_{coil} = \mathbf{A}\_\varphi_{coil} \cdot \mathbf{A}. \tag{4.25}$$

The above process establishes a 6-by-6 current-to-voltage mapping between the coils. Thus the transformer FE system becomes a 6-port nonlinear current ($\mathbf{I}^t$)-controlled voltage source ($\mathbf{V}^t$), and the Jacobian matrix of the nonlinear source is obtained by similar small

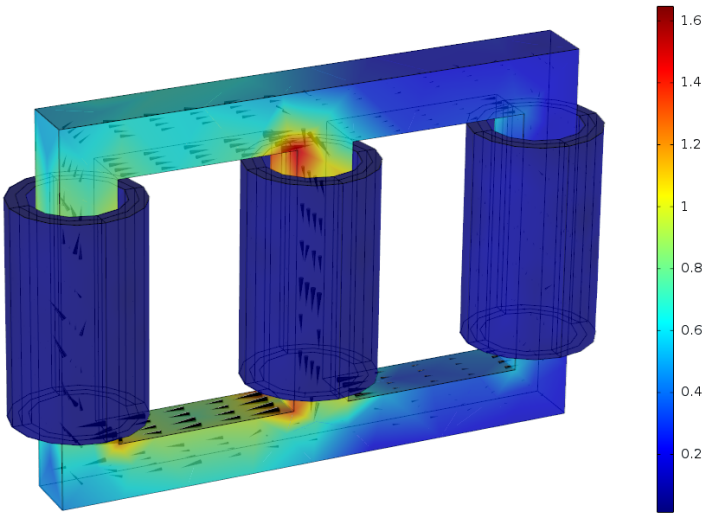


Figure 4.11: Magnetic flux density (B) field of 3-phase transformer at 0.00733s.

Table 4.3: Transformer Problem Definition

Transformer parameters					
Coil/air permeability	μ_0	Winding turns primary/secondary	750/1500	Core σ	2000S
Iron core B-H curve	H= $\begin{cases} v_0 B/1000, \ \|B\| \leq 1.3 \\ v_0 B/1000 + 1e5(\|B\| - 1.3)^3 \times \|B\|/B, \ \|B\| > 1.3 \end{cases}$				
Circuit parameters					
$R_s = 15\Omega$		$R_p = 300\Omega$		$C = 300nF$	
V_{pow} are sinusoidal waves with peak amplitude 60kV at 60Hz					
Simulation time set-up					
Time-step length: 1/60/200s			Total time span: 0 to 0.15s		

probing increment separately superimposed on 6 coils:

$$\frac{\partial \varphi_{\Delta t}}{\partial I^t}_{ij} = \frac{\varphi_{\Delta t}\left(\mathbf{I}^t\right)_i - \varphi_{\Delta t}\left(\mathbf{I}^t \cup dI^t{}_j\right)_i}{dI^t{}_j}, i, j \in [1, 6], \tag{4.26}$$

where $dI^t{}_j$ means small probing increment of current on coil j, and $\varphi_{\Delta t}\left(\mathbf{I}^t \cup dI^t{}_j\right)_i$ means time-discretized flux on coil i, generated form reference current $\mathbf{I}^t$ superimposed with $dI^t{}_j$.

The extended FEM model is connected to the balanced 3-phase circuit shown Fig. 4.10. The circuit switch simulates an open-circuit fault at the secondary loads R_s. The fault starts at 0.05s with a duration of 0.05s.

Based on the model in Fig. 4.10 and parameters in Table 4.3, the simulation was carried out at a fixed mesh size of 15006 DoFs. During computation, the iterations needed for each time-step are plotted in Fig. 4.13, and time consumed is 501s vs. 1198s on $Comsol^{TM}$. The

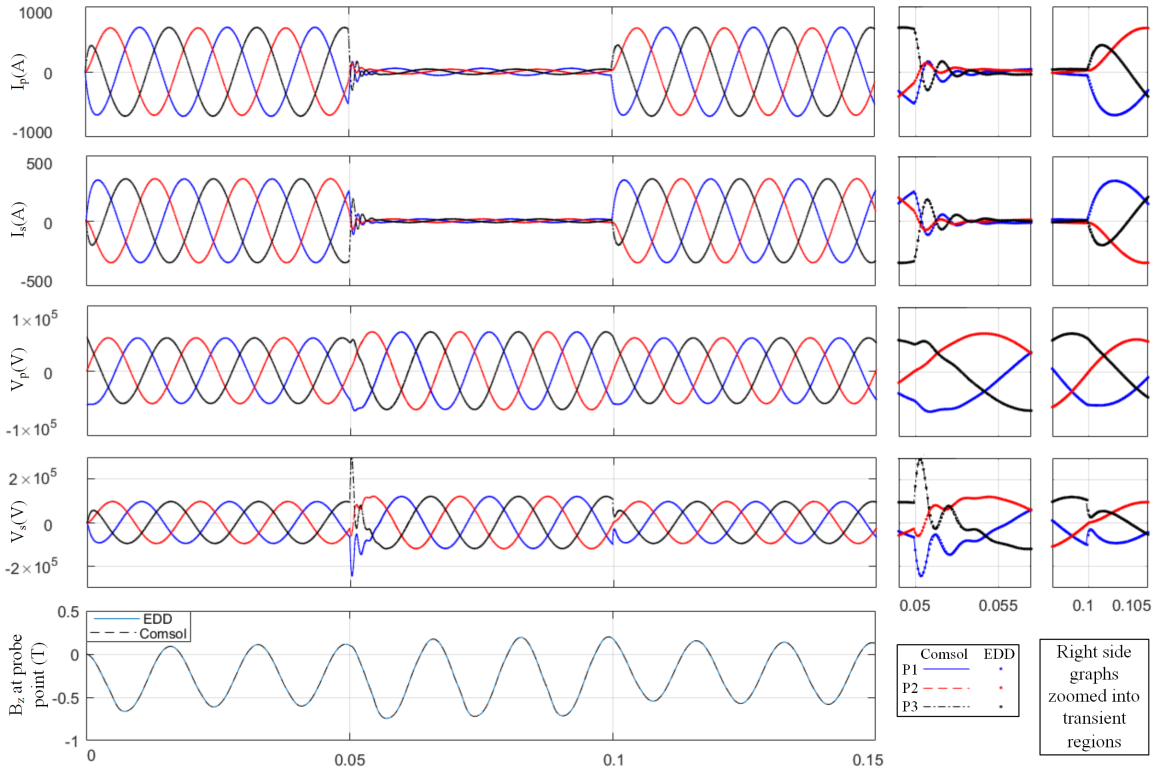


Figure 4.12: Comparison of EDD scheme and $Comsol^{TM}$ results over time (s) for the 3-phase transformer.

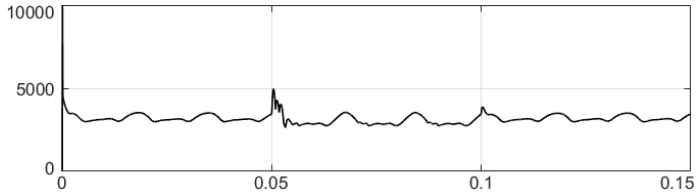


Figure 4.13: Total EDD iterations vs. time (s) with field-circuit interfaceing. Note: circuit-field iteration count is fixed at 5 for all time-steps. The figure gives the sum of all coil-parallel EDD iterations involved in 5 coil-field iterations.

final result is shown in Fig. 4.11-4.12 and a comparison with $Comsol^{TM}$ shows a relative error of less than 2% over time and space domain.

4.7 Discussion about Speed-up of EDD Scheme

As is shown in Table 4.2, the speed-up decreases with the mesh DOFs, which leads to the following question: will the EDD scheme becomes slower than $Comsol^{TM}$ for millions of DOFs? The current performance is hampered by total GPU cores of 5120. This means many sub-domain solvers are still sequentially executed. However, if all sub-domains are solved in parallel, the time needed will be proportional to the number of EDD iterations, which

is slower than the growth of mesh DOFs, and this results in an increasing speed-up vs. DOFs compared to $Comsol^{TM}$ (note that in reality, there might be some implementation limit such as communication delay between large number of cores).

It is also seen that $Comsol^{TM}$ is much faster (such as 50 times) in time-domain compared with the static scenario. This is possibly because it utilizes a pre-factorize-and-back-substitution method to save time at the linear time-steps. The system matrix may be factorized only once and only light back-substitutions are carried out at time-steps of linear material B-H region. Thus, the time consumption is not comparable with the EDD-circuit scheme, since the method does both 'factorization and back-substitution' at each time-step.

Future research will focus on expending the EDD scheme on clusters with multiple GPUs to exceed the 5000-core limit. Also, we will integrate the EDD scheme with the pre-factorize-and-back-substitution method to gain better speed-up for time-domain cases.

4.8 Parallel-in-Time Method for 3-D Finite Edge Elements

4.8.1 Extending Parallelism in the Time Dimension

Algorithms in the above discussion, either the decentralized edge domain decomposition or the constant-global elemental TLM method, are based on space parallelism. Also, they are mostly implemented on a single GPU. To further improve parallelism for large-scale hardware such as GPU/CPU clusters, this section explores another possibility to parallelize 3-D FEM models with the time dimension. The Parareal parallel-in-time technique suitable to solve ordinary differential equations is firstly introduced. Then, the 3-D FEM system was combined with a parallel algorithm in a case study with a decent speed-up.

4.8.2 Parareal Method for the Solution of Ordinary Differential Equations of FEM-Circuit System

The Parareal algorithm can accelerate time-domain ordinary differential equations (ODE) by introducing parallelism along the time span, and the method has been applied to nonlinear circuit solvers [41] and even nonlinear 2-D FEM analysis [42] with decent speed-ups, since they are essentially ODE systems. On the other hand, due to the inherent ODE nature, the TLM field solver or EDD field-circuit solver in the above chapters can also be integrated with the Parareal method. As a result, the parallelism is further extended into both time and space levels so that hardware resources can be fully exploited.

To explain how the space-time parallelized field-circuit solver works, we introduce the following definitions:

1. Each edge's magnetic potential A^t, and the circuit's all nodal voltages and branch currents are noted as state variables. And these state variables are packed in one vector U_t.

2. By repeating the algorithm in Fig. 4.7, the state variables can be propagated from the initial time point t_1 to a given future time point t_2. When the system time step Δt is small

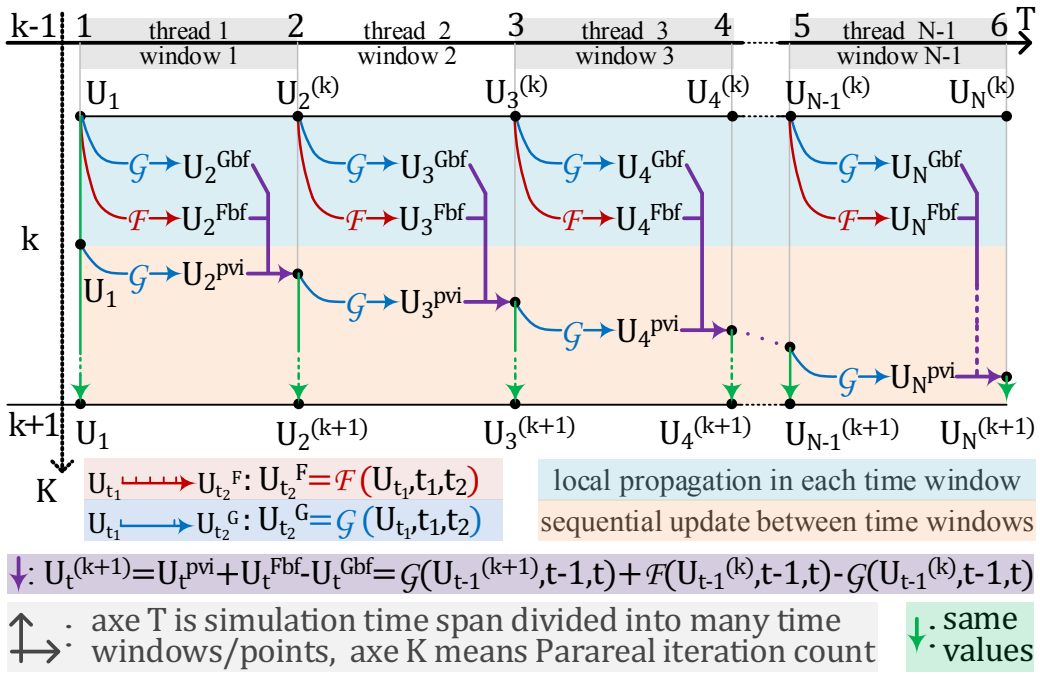


Figure 4.14: Parareal algorithm explained in detail. Note: U_2~U_N are all set to 0 at the start of Parareal iteration. And U_1 is the known initial boundary value.

enough, the program yields precise state variable value at t_2, and the computation process is defined as the precise propagator $\mathcal{F}$:

$$U_{t_2}{}^{F} = \mathcal{F}\left(U_{t_1}, \mathrm{t}_1, \mathrm{t}_2\right), \tag{4.27}$$

If the time step Δt is much bigger compared to $\mathcal{F}$, the program in Fig. 4.7 can only generate a coarse estimation of state variables at t_2 with a much cheaper computation amount. This process is noted as the coarse propagator $\mathcal{G}$:

$$U_{t_2}{}^{G} = \mathcal{G}\left(U_{t_1}, \mathrm{t}_1, \mathrm{t}_2\right), \tag{4.28}$$

Due to the time step difference, $\mathcal{G}$ requires light computation while $\mathcal{F}$ becomes computationally expensive.

As is shown in Fig. 4.14, the Parareal algorithm divides the whole time span into N-1 smaller time windows. Between these time windows are the time points where the state variables need to be found. Traditionally, to achieve precise results at these time points, operator $\mathcal{F}$ needs to sequentially propagate state values between each time window, which could be time-consuming considering the dependency of later time points on previous ones. In contrast, the Parareal algorithm executes the propagators inside each time window in parallel and utilizes iterations to make the values at each time point converge to the precise result.

During each iteration, every time window is assigned to a working thread. Each time window propagates in parallel the state values from their local start time point to the local

endpoint, and the result is saved in their local buffers (${U_i}^{Gbf}$ or ${U_i}^{Fbf}$). Note that these local propagations involve both $\mathcal{F}$ and $\mathcal{G}$, and the computational expensive $\mathcal{F}$ is parallel-executed in each different time windows with different initial values. After the local propagations, state values at each time point are sequentially updated by the local buffers and the information from the previous time point as follows:

$$ {U_t}^{(k+1)} = \mathcal{G}\left({U_{t-1}}^{(k+1)}, \mathrm{t}-1, \mathrm{t}\right) + \mathcal{F}\left({U_{t-1}}^{(k)}, \mathrm{t}-1, \mathrm{t}\right) - \mathcal{G}\left({U_{t-1}}^{(k)}, \mathrm{t}-1, \mathrm{t}\right), \qquad (4.29) $$

where k is the Parareal iteration count, and t is the time point to be updated.

The Parareal iteration repeats until state variables at the last time point converge into a stable result. If the Parareal iterations required are less than the number of time windows, the parallel-in-time algorithm becomes faster than the traditional time-serial method. Once the Parareal algorithm is combined with the space-parallelized finite element method in the above chapters, inductive power devices can be accelerated with both space and time level parallelism at the time points between the time windows.

4.8.3 Case Study for Time-Parallel Technique with 3-D Edge Finite Elements

To verify the feasibility and efficiency of Parareal technique in 3-D edge finite elements, the above algorithm is applied to the FEM field-circuit system in 4.6.2. The Parareal case study carries out based on the same circuit, transformer, fault configuration, and total time-span (0.15s). The Parareal algorithm divides the entire time-span into 15 smaller time windows for the implementation. Within each time window, the propagator $\mathcal{G}$ is assigned a big time-step of 0.1s for coarse prediction, and the propagator $\mathcal{F}$ is given a much smaller time-step (0.005s) to precisely calculate the state variables.

After computation, the results indicate that the Parareal algorithm takes 5 iterations to converge with a relative error of less than 1% vs. its time-serial counterpart. Since the entire computation can be assigned to 15 different threads, the theoretical speed-up becomes 15/5=3 for this case study.

4.9 Summary

In this chapter, a novel edge domain decomposition method was proposed to calculate the 3-D field for nonlinear electromagnetic devices. The algorithm achieved massive parallelism and was implemented on GPU architectures.

The idea of minimum sub-domain division is implemented with time-domain 3-D nonlinear edge elements based on RMVP formulation. Benefitting from the extreme sub-domain size, the method simultaneously solves material nonlinearity and global FE system without having to assemble the global matrix. Also, the light-weight sub-domain task and huge sub-domain number result in excellent modularity and massive parallelism, which made the EDD scheme a perfect choice for many-core GPU implementation. Furthermore,

an auto-gauging property was seen to converge the rank-deficient system into a stable solution.

For field-circuit co-simulation, a coupling scheme was developed to interface the EDD scheme with the external circuit. The scheme can abstract away complex FE model in controlled sources, while maintaining a high precision at the same time, especially under high eddy current conditions. The efficiency and accuracy of the EDD-circuit method were discussed and verified through the GPU implementation. The comparison with $Comsol^{TM}$ indicates a significant speed-up of 43.7 with an error of less than 2%.

Also, the Parareal method was explored to increase parallelism as much as possible at the space-time dimension. The simulation of the EDD field circuit system was executed in 15 separate time windows and the result indicates a theoretical speed-up of over 3. Thus, the Parareal algorithm can be another effective way to boost 3-D edge finite element models in power system simulation.

5 Conclusion

5.1 Summary of the Thesis

The decentralized idea of nodal domain decomposition and the transmission line decoupling method were successfully implemented with 3-D edge finite element models based on reduced magnetic vector potential formulation. Parallelism was achieved at different levels with elements, edges, and time. These algorithms were explored, developed, and tested with decent efficiency. The summary of contributions is given below:

- The transmission line decoupling technique was extended from circuit analysis into 3-D edge FEM models, and parallelism was achieved at the tetrahedron elemental level. The method converts the 3-D nonlinear matrix system into a big-scale circuit network with both linear and nonlinear resistors. The nonlinear resistors associated with each element were decoupled from the big-scale global linear resistor network with transmission lines. This resulted in a constant global matrix (linear resistors) and many isolated small-scale local nonlinear matrixes (elemental nonlinear resistors). Since the global matrix is constant, it only requires only one factorization during the entire computation. And each elemental nonlinear matrix was assigned to a GPU thread to achieve massive parallelism. The local systems and the global matrix communicates and a consistent solution can be reached in the end. The TLM decoupled edge element model was tested on the Tesla V100 GPU, and the result shows excellent speed-up (over 50) and accuracy (2% error) comparing with $Comsol^{TM}$.

- The edge domain decomposition (EDD) algorithm was created to boost the computation of 3-D finite element models with parallelism at edge level. The decentralized idea of dividing the entire solution domain into many minimum overlapped sub-

domains was combined with 3-D finite edge tetrahedron elements. Such overlapping nature and the minimum size of the sub-domain leads to an advanced iteration scheme that eliminates the need to form any global matrix during the entire solution of the nonlinear finite element problem. Also, benefiting from the light-weight size of the sub-domains problems, this matrix-free iteration scheme became perfectly suitable for implementation over the GPU's single instruction multiple data architectures. The GPU-friendly and matrix-free EDD scheme was also developed and tested with CUDA C language on Tesla V100 GPU over different mesh sizes. And the comparison with $Comsol^{TM}$ indicates a significant speed-up (best case over 43) and a great potential for large-scale FEM problems of million-level degree of freedoms.

- An indirect coupling scheme was proposed to interface the FEM inductive device model and the circuit that dives the device. The complex finite element system was packaged into a current-controlled voltage source for the convenience of circuit system computation. And a field-circuit iterative scheme was developed to obtain a consistent solution between the circuit and FEM systems. Due to the package-and-iterative nature, the coupler does not require the combination of FEM and circuit matrixes, allowing the co-simulation of FEM models and large-scale complex circuit systems. The coupler was implemented in a 3-phase transformer-circuit study and the result indicates a fast convergence (typically less than 5 field-circuit iterations) without compromising precision vs. the traditional direct mixed matrix coupling scheme.

- The thesis also explored another possibility to expand parallelism from time's perspective. The parallel-in-time or the Parareal algorithm was introduced and adapted to the above space-parallelized 3-D edge finite element system. For the first time, the space-time parallelism was achieved in 3-D edge finite element models, and the implementation indicated an additional speed-up of 3, given enough parallel computing hardware resources.

5.2 Suggestions for Future Work

The algorithms of the previous chapters can be improved through the following aspects:

- The coils of the above FEM transformer were based on homogeneous multi-turn technology. Future research can focus on distributed modeling of the coils to provide more realistic results.

- The back-substitution process caused the main time consumption in the TLM decoupled 3-D edge finite element models. Better parallelized back-substitution packages can be applied and tested to improve the total efficiency of the TLM algorithm.

- The edge domain decomposition technique was based on the first type of boundary conditions to update the sud-domain central edge DOF. And most information from the sub-domain matrix was discarded. Therefore, the second type of boundary may be applied to fully utilize information of the sub-domain matrix, which can possibly lead to faster convergence speed.
- All of the above methods are developed for a single high-performance GPU. And speed-up is hindered due to a limited number of cores for large problem sizes. This means that the algorithms can be implemented on larger platforms (such as super-computers) to achieve faster computation speed.
- Due to the matrix-free property, the EDD scheme can be applied in changing geometry meshes without additional performance loss. This provides the possibility to achieve high-performance simulation for electrical machines in the future.

Bibliography

[1] J. A. Martinez and B. A. Mork, "Transformer modeling for low- and mid-frequency transients - a review," *IEEE Trans. on Power Delivery*, vol. 20, no. 2, pp. 1625–1632, 2005.

[2] V. Brandwajn, H. W. Donnel, and I. I. Dommel, "Matrix representation of three-phase n-winding transformers for steady-state and transient studies," *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-101, no. 6, pp. 1369–1378, 1982.

[3] F. de Leon and A. Semlyen, "Complete transformer model for electromagnetic transients," *IEEE Trans. on Power Delivery*, vol. 9, no. 1, pp. 231–239, 1994.

[4] N. D. Hatziargyriou, J. M. Prousalidis, and B. C. Papadias, "Generalised transformer model based on the analysis of its magnetic core circuit," *IEE Proceedings C - Generation, Transmission and Distribution*, vol. 140, no. 4, pp. 269–278, 1993.

[5] A. Narang and R. H. Brierley, "Topology based magnetic model for steady-state and transient studies for three-phase core type transformers," *IEEE Trans. on Power Systems*, vol. 9, no. 3, pp. 1337–1349, 1994.

[6] J. Arrillaga, W. Enright, N. R. Watson, and A. R. Wood, "Improved simulation of hvdc converter transformers in electromagnetic transient programs," *IEE Proceedings - Generation, Transmission and Distribution*, vol. 144, no. 2, pp. 100–106, 1997.

[7] J. Liu and V. Dinavahi, "A real-time nonlinear hysteretic power transformer transient model on fpga," *IEEE Trans. on Industrial Electronics*, vol. 61, no. 7, pp. 3587–3597, 2014.

[8] ——, "Nonlinear magnetic equivalent circuit-based real-time sen transformer electromagnetic transient model on fpga for hil emulation," *IEEE Trans. on Power Delivery*, vol. 31, no. 6, pp. 2483–2493, 2016.

[9] S. J. Salon, *Finite element analysis of electrical machines*. Kluwer academic publishers Boston USA, 1995, vol. 101.

[10] [Online]. Available: https://www.computecanada.ca/techrenewal/

[11] [Online]. Available: https://images.nvidia.com/content/technologies/volta/pdf/tesla-volta-v100-datasheet-letter-fnl-web.pdf

[12] Y. Li and J. Jin, "A vector dual-primal finite element tearing and interconnecting method for solving 3-d large-scale electromagnetic problems," *IEEE Trans. on Antennas and Propagation*, vol. 54, no. 10, pp. 3000–3009, 2006.

[13] [Online]. Available: https://en.wikipedia.org/wiki/Domain_decomposition_methods

[14] O. Schenk, K. Artner, G. Karypis, and S. Ollin, "Pardiso solver project," *URL: http://www.pardiso-project.org*, 2010.

[15] L. Ziane Khodja, R. Couturier, A. Giersch, and J. Bahi, "Parallel sparse linear solver with gmres method using minimization techniques of communications for gpu clusters," *The Journal of Supercomputing*, vol. 69, 03 2014.

[16] P. Liu and V. Dinavahi, "Matrix-free nodal domain decomposition with relaxation for massively parallel finite-element computation of em apparatus," *IEEE Trans. on Magnetics*, vol. 54, no. 9, pp. 1–7, 2018.

[17] ——, "Real-time finite-element simulation of electromagnetic transients of transformer on fpga," *IEEE Trans. on Power Delivery*, vol. 33, no. 4, pp. 1991–2001, 2018.

[18] J.-M. Jin, *The finite element method in electromagnetics*. John Wiley & Sons, 2015.

[19] M. Barton and Z. Cendes, "New vector finite elements for three-dimensional magnetic field computation," *Journal of Applied Physics*, vol. 61, no. 8, pp. 3919–3921, 1987.

[20] J. P. A. Bastos and N. Sadowski, *Magnetic materials and 3D finite element modeling*. CRC press, 2013.

[21] J. Li, P. Liu, and V. Dinavahi, "Matrix-free edge domain decomposition method for massively parallel 3-d finite element simulation with field-circuit coupling," *IEEE Trans. on Magnetics*, pp. 1–1, 2020.

[22] P. B. Johns and M. O'Brien, "Use of the transmission-line modelling (t.l.m.) method to solve non-linear lumped networks," *Radio and Electronic Engineer*, vol. 50, no. 1.2, pp. 59–70, January 1980.

[23] C. Christopoulos, *The transmission-line modeling method: TLM*. IEEE New York, 1995, vol. 221.

[24] J. Lobry, J. Trecat, and C. Broche, "The transmission line modeling (tlm) method as a new iterative technique in nonlinear 2-d magnetostatics," *IEEE Trans. Magn.*, vol. 32, no. 2, pp. 559–566, March 1996.

[25] 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.

[26] P. Liu, J. Li, and V. Dinavahi, "Matrix-free nonlinear finite-element solver using transmission-line modeling on gpu," *IEEE Trans. on Magnetics*, vol. 55, no. 7, pp. 1–5, 2019.

[27] A. Demenko, J. K. Sykulski, and R. Wojciechowski, "On the equivalence of finite element and finite integration formulations," *IEEE Trans. Magn.*, vol. 46, no. 8, pp. 3169–3172, Aug 2010.

[28] A. Demenko and J. K. Sykulski, "Geometric formulation of edge and nodal finite element equations in electromagnetics," *COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, vol. 31, no. 5, pp. 1347–1357, 2012.

[29] R. Albanese and G. Rubinacci, "Finite element methods for the solution of 3d eddy current problems," in *Advances in Imaging and Electron Physics*. Elsevier, 1997, vol. 102, pp. 1–86.

[30] Y. Zhao and W. N. Fu, "A novel formulation with coulomb gauge for 3-d magnetostatic problems using edge elements," *IEEE Trans. Magn.*, vol. 53, no. 6, pp. 1–4, June 2017.

[31] ——, "A novel coulomb-gauged magnetic vector potential formulation for 3-d eddy-current field analysis using edge elements," *IEEE Trans. Magn.*, vol. 53, no. 6, pp. 1–4, June 2017.

[32] A. Joseph, A. Joseph, and Administer, *Theory and Problems of Electric Circuits*. Mc Graw Hill., 1994.

[33] B. Asghari, V. Dinavahi, M. Rioual, J. A. Martinez, and R. Iravani, "Interfacing techniques for electromagnetic field and circuit simulation programs ieee task force on interfacing techniques for simulation tools," *IEEE Trans. on Power Delivery*, vol. 24, no. 2, pp. 939–950, 2009.

[34] O. Widlund and M. Dryja, *Towards a unified theory of domain decomposition algorithms for elliptic problems*, ser. Technical Report 486, Ultracomputer Note 167. Department of Computer Science, Courant Institute, 12 1989.

[35] C. T. Wolfe, U. Navsariwala, and S. D. Gedney, "A parallel finite-element tearing and interconnecting algorithm for solution of the vector wave equation with pml absorbing medium," *IEEE Trans. on Antennas and Propagation*, vol. 48, no. 2, pp. 278–284, 2000.

[36] J. Li, P. Liu, and V. Dinavahi, "Massively parallel computation for 3-d nonlinear finite edge element problem with transmission line decoupling technique," *IEEE Trans. on Magnetics*, vol. 55, no. 10, pp. 1–8, 2019.

[37] I. Kiss, S. Gyimothy, Z. Badics, and J. Pavo, "Parallel realization of the element-by-element fem technique by cuda," *IEEE Trans. on Magnetics*, vol. 48, no. 2, pp. 507–510, 2012.

[38] J. P. A. Bastos and N. Sadowski, "A new method to solve 3-d magnetodynamic problems without assembling an $ax = b$ system," *IEEE Trans. on Magnetics*, vol. 46, no. 8, pp. 3365–3368, 2010.

[39] R. Escarela-Perez, E. Melgoza, and J. Alvarez-Ramirez, "Coupling circuit systems and finite element models: A 2-d time-harmonic modified nodal analysis framework," *IEEE Trans. on Magnetics*, vol. 45, no. 2, pp. 707–715, 2009.

[40] N. T. K Fujiwara and T. Nakata, *Advanced Computational and Design Techniques in Applied Electromagnetic Systems*, ser. Technical Report 486, Ultracomputer Note 167. Department of Computer Science, Courant Institute, 06 1995.

[41] T. Cheng, T. Duan, and V. Dinavahi, "Parallel-in-time object-oriented electromagnetic transient simulation of power systems," *IEEE Open Access Journal of Power and Energy*, vol. 7, pp. 296–306, 2020.

[42] S. Schöps, I. Niyonzima, and M. Clemens, "Parallel-in-time simulation of eddy current problems using parareal," *IEEE Transactions on Magnetics*, vol. 54, no. 3, pp. 1–4, 2018.