HIGHLY PARALLEL ALGORITHMS AND SYSTEMS FOR FAST ELECTROMAGNETIC TRANSIENT SIMULATION IN POWER SYSTEM

A Thesis

by

LU ZHANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Weiping Shi
Committee Members,	P. R. Kumar
	Le Xie
	Anxiao (Andrew) Jiang
Head of Department,	Miroslav M. Begovic

May 2021

Major Subject: Computer Engineering

ABSTRACT

The significant increase of variable energy resources in the power grid, coupled with the substantial growth of electrified vehicles, leads to a more stressed grid with much higher variabilities at the operational stage. Such variabilities together with today's lack of accurate simulation capabilities lead to significant uncertainties in predicting the dynamics across the grid, which when combined with lower operating reserves lads to dangerous combination with respect to grid reliability. To address this requires a capability to accurately predict the future dynamic behavior of the grid in a faster than real-time manner for the range of uncertainties, taking all factors, electromechanical, electrical and electromagnetic, into account. Among all of the transient simulations, electromagnetic transient (EMT) simulation is the most powerful tool which could handle various detailed device models and capture very high speed dynamic behavior in the power system. Thus, simulating the EMTs in a faster than real-time, and high accuracy manner is highly demanded. In this thesis, we propose a computational framework to model, convert, and accelerate the grid-level EMT simulations based on a highly parallel inverse-based low-rank approximation approach which is suitable for full-custom VLSI (very large scale integration) techniques.

The first technique we propose is to tailor various models of the key elements in power system into a general wide-band model and break down the EMT problem into a hardware friendly framework for ASIC architecture. Based on our best knowledge, we adopt the nodal analysis formulation to simulate the EMT problem, which could map multiple device models into the framework.

The second technique we propose is to tackle the bottleneck of EMT simulation - network solution which takes 80-97% of the computational time. Traditional approaches to solve network equations are based on sparse LU factorization, which is inherently sequential. We propose a highly parallel inverse-based network solution based on hierarchical low-rank approximation which permits $O(N \log N)$-time matrix-vector multiplication for each network solution time step. Comprehensive numerical studies are conducted on a 39-bus system and a 179-bus system from the literature, and large cases are created from the two systems. The results demonstrate that the

proposed approach is up to $2.8\times$ faster than the state-of-the-art sparse LU factorization based network solution, without compromising simulation accuracy. Since our low-rank approximation is highly parallelizable, further speedup can be realized by hardware accelerators.

The third technique we propose is to reduce the time cost of computing, storing, and updating the inverse of the large sparse conductance matrix in EMT. The inverse-based network solution shows a great benefit in speeding up the EMT simulation with the low-rank approximation approach. However, the computation and storage of the inverse of the conductance matrix will be very expensive especially when a fault occurs. We propose a hierarchical computation and modification method which can not only efficiently compute and storage the inverse of the conductance matrix, but also update the inverse by modifying only local sub-matrices to reflect changes in the network, e.g., loss of a line. Experiments on a series of simplified 179-bus Western Interconnection demonstrate the advantages of the proposed methods.

The fourth technique we propose is to develop a full-custom application-specific integrated circuit (ASIC) that can be leveraged for accelerating the EMT simulation in power systems based on the class of parallel algorithms we proposed. Numerical results show that the proposed ASIC architecture can simulate a 2-bus system significantly fast. Large-scale optimal ASIC design will be developed by Naga Shiva Sai Pavan Kumar Devarasetti, a master student of Dr. Weiping Shi.

DEDICATION

To my husband, Jizhou Fan,

for being with me, supporting me, and experience everything together.

To my parents,

for their unconditional love and support.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Weiping Shi, for his endless support, encouragement, and guidance throughout the course of my Ph.D. study. I appreciate that he gladly accepted me to take my first step for becoming a researcher under his guidance. His patience allowed me to make steady and consistent progress. Whenever I was stuck on a problem, he has been always with me to give precious advice to help me moving forward. Without his help, I would not have completed this work.

I would also like to thank my committee members, Dr. P. R. Kumar, Dr. Le Xie, and Dr. Anxiao Jiang for their insightful comments and support.

In addition, I would like thank Dr. Vivek Sarin for his insightful idea and inspiration which contributes to Chapter 4.

My thanks also go to Dr. Bin Wang, Dr. Meng Wu, Zhixing Li, Xiangtian Zheng, Dongqi Wu, Naga Shiva Sai Pavan for gladly taking their time for discussion and collaboration.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

All work conducted for the thesis (or) dissertation was completed by the student, under the advisement of Weiping Shi of the Department of Electrical and Computer Engineering.

Funding Sources

Graduate study was supported in part by a fellowship from Texas A&M University, in part by NSF OAC-1934675, in part by ECCS-1760554, in part by CCF-0939370, in part by ECCS-1611301, in part by ECCS-1839616, and in part by the Power Systems Engineering Research Center.

NOMENCLATURE

EMT	Electromagnetic Transients
SSO	Sub-Synchronous Oscillations
TS	Transient Simulation
DAE	Differential-Algebraic Equation
ODE	Ordinary-Differential Equation
FPGA	Field-Programmable Gate Array
CPU	Central Processing Units
GPU	Graphics Processing Units
VLSI	Very Large Scale Integration
ASIC	Application-Specific Integrated Circuit
EDA	Electronic Design Automation
HODLR	Hierarchically Off-Diagonal Low Rank
SVD	Singular Value Decomposition
SLU	Sparse LU Factorization
HiLap	Hierarchical Low-rank Approximation
FLOPS	Floating Point Operations Per Second
LCA	Lowest Common Ancestor
IC	Integrated Circuits

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Background

In modern power systems, the significant increase of variable energy resources such as renewable generations in the power grid, coupled with the substantial growth of electrified vehicles, leads to a more stressed grid with much higher variability at the operational stage. Such variability together with today's lack of accurate online simulation capabilities lead to the difficulty in predicting the transient dynamics across the grid. This, combined with lower operating reserves, pose significant challenges to online security assessment. To address these issues, simulations with a capability to accurately predict the future transient behavior of the grid in a faster than real-time manner, are taking all factors, such as electromechanical and electromagnetic, into account. The most acceptable current practices when there were sufficient operating reserve buffers in the past are only considering electromechanical transients which are already slow, and insufficient for the next-generation grid transformed by the reliance on many more variable resources and serving many more varying demands, with a much lower reserve margin [1, 2, 3]. Fig. 1.1 shows an incomplete list of growing physical anomaly partially induced by electromagnetic transients (EMTs) over the past five decades. Thus, to ensure the safety and security of individual elements and the entire power system, simulating the EMTs in a faster than real-time, and high accuracy manner is highly demanded in many designs and studies, e.g., the component rating design, insulation coordination, lightning overvoltage computation, HVDC converter control and operation, and the design of protection and control [4]. Besides that, EMT simulation is also required when studying new problems such as geomagnetically-induced currents [5].

Different from transient dynamic analysis, EMT is capable to simulate dynamic phenomena of extremely various generators, transmission lines, transformers, and other elements in a wide range of frequencies from DC to 50MHz in the power system. This requires EMT to handle amounts of detailed models to capture the dynamics which requires expensive computations. Many tools have

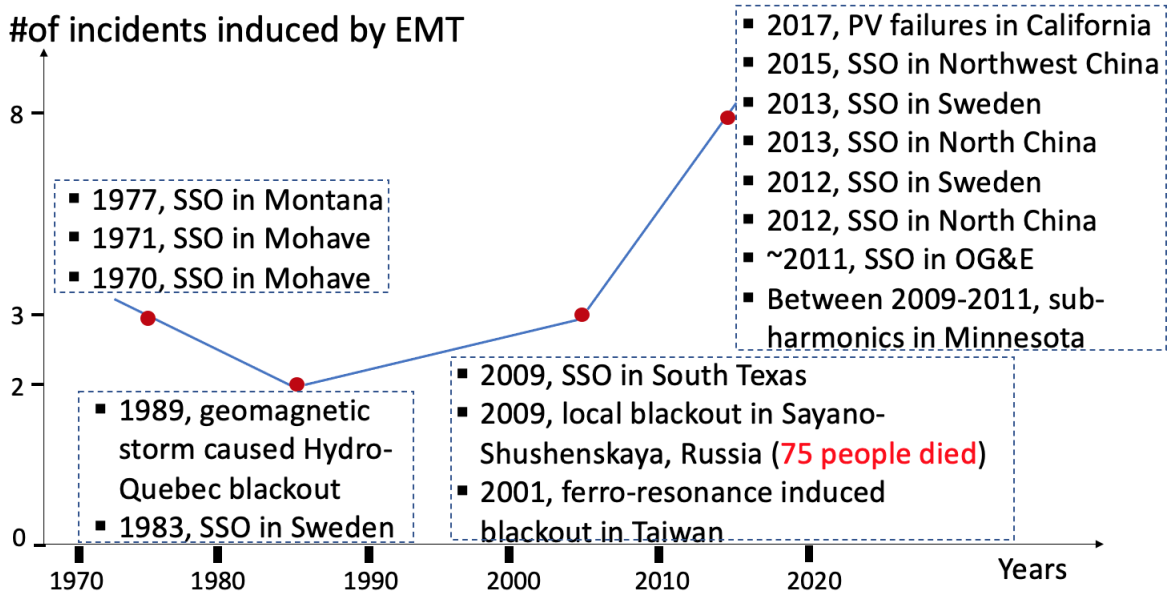


Figure 1.1: EMT-related incidents in the power grid where SSO (sub-synchronous oscillations) is the main concern.

been developed for EMT simulation [6, 7, 8] and extensive research efforts have been devoted to achieve further speedup by, e.g., reduction of the power system model [9], development of more efficient element model [10], utilization of dedicated computing devices [11], parallelism [5] and combinations of the above [12].

In the power industry, EMT is typically performed through two types of commercial simulation tools. The first type of tools is software based which focuses on the offline EMT simulation such as PSCAD [13] and EMT-RV [14]. The second type of tools focuses on real-time digital simulators such as RTDS [15] and OPAL-RT [16]. Among the simulators, most of them [16] rely on hardware like field-programmable gate array (FPGA) or GPUs to realize the real-time simulation. Taking the advantage of high computational performance, the hardware-based simulators are shown to have much faster compared to software-based simulators. Unfortunately, state-of-art real-time simulation can only handle small-scale systems, while the simulation of large-scale systems can only be done in an offline environment. The so called 'real-time' simulation is still not fast enough to realize the true real-time in a large system which means that the simulation of one second in the

power system takes more than one second. The state-of-the-art commercial EMT simulator takes up to 80 hours to simulate one second of transient for an $87,000$-node system [17] which is far away from real-time.

1.2 Overview of Our Work

In this dissertation, we aim to develop a fast EMT simulation by high parallel algorithms which could further speed up by hardware accelerators such as very large scale integration (VLSI). To achieve the practical problem, we address it into four sub-problems.

The first step is to deeply understand the EMT simulation framework and use a practical approach to implement the offline EMT simulation w/wo fault in the software, e.g., MATLAB and C, then compare our implementation with commercial software, e.g., EMT-RV to make sure we are in the same page. Based on our accurate software implementation, we optimize the simulation flow, target the bottleneck and reorganize it so that it could map into a parallelizable hardware design.

Second and the most important problem - speed up the network solution. Among all approaches in the EMT simulation, the network solution is often the most time-consuming part, which takes $80-97\%$ [18, 19] of the entire time cost. Thus, the network solution is a bottleneck of the EMT simulation. In practice, there are three typical types of approaches to solving the network equations, i.e. (1) dense inversion and matrix-vector products, (2) direct dense solver and (3) direct sparse solver. Among them, sparse LU factorization-based direct sparse solver exhibits the most efficient for systems with $> 2K$ dimensions [12]. This explains why sparse LU is the most widely used solver for network equations in commercial EMT simulation software. However, its performance is insufficient to meet the challenge, and LU-based solvers are sequential in nature and difficult to parallelize for additional speedup. Although if any network delay exists, it can be naturally used to decompose the network to achieve a kind of parallelization, the resulting improvement is very limited when there exist large blocks, e.g., of greater than 13K dimensions, representing dense regions [20]. In addition, few work has been focused on how to further decompose the network [21].

In this dissertation, we propose an inverse based highly parallelizable approach to accelerate

EMT simulation by taking advantage of the network topological structure and achieve fast matrix-vector multiplication by low-rank approximation. The low-rank approximation significantly reduces the computational time and speeds up the network solution to $O(N \log N)$ outperforming previous network solvers, including sparse LU based solvers. We start from restating the network solution to invert-based format, then in the offline, hierarchically partition the entire network into multiple sub-networks based on the interaction strength between buses, so that we could apply the low-rank approximation hierarchically to approximate the interaction between sub-networks. To highly reduce the online computation cost, we propose a fast matrix-vector multiplication approach based on the low-rank approximation results, and apply it in each simulation time step. To demonstrate the benefit of the proposed approach, we present the time complexity, memory and error analyses. We also conduct extensive performance tests and compare with the state-of-the-art sparse LU-based network solver.

Our third project focuses on further speed up network solution by fast conductance matrix inversion computation and modification. As we mentioned above, the traditional sparse LU-based approaches inherently sequential which is not friendly for hardware acceleration. In contrast, the inverse-based approach exhibits great potential in speeding up the network solution by parallel matrix-vector multiplication. However, direct compute sparse matrix inverse is very expensive and inefficient when handling the scenario where matrix changes, especially when the system is very large. Thus, we propose a hierarchical fast and memory-efficient approximation method for computing and modifying the inverse of the large admittance/conductance matrices established from power grids. The new method can efficiently update the inverse matrix by modifying only local entries and sub-matrices to reflect changes in the network, e.g., loss of a line. The advantage of the proposed method is demonstrated on both small and large system EMT simulation, similar results can be expected in transient stability simulation as well.

The fourth problem we investigate is to implement the proposed algorithms on computation efficient hardware to accelerate the simulation at the hardware level. For EMTP simulation, hardware-accelerated methods using FPGAs are much faster than software methods using cen-

tral processing units (CPUs). However, the FPGA approach has two deficiencies: 1) The proposed FPGA approaches use mostly traditional numerical algorithms, thus fail to fully exploit parallelism available in digital circuits; and 2) FPGA architectures are for general purpose application, thus not optimal for EMTP simulation. Thus, we introduce and test a first-of-its-kind framework that builds customized and highly parallel VLSI (very large scale integration) hardware architecture by advanced EDA tool in order to substantially improve the speed of EMTP simulation for large power systems.

1.3 Outline

Compared with the traditional approaches, the major contribution of the dissertation is presented a novel parallelizable approach for the fast EMT simulation in large scale power system in terms of efficiency computation cost, high accuracy and strong scalability, shown in Fig. 1.2.

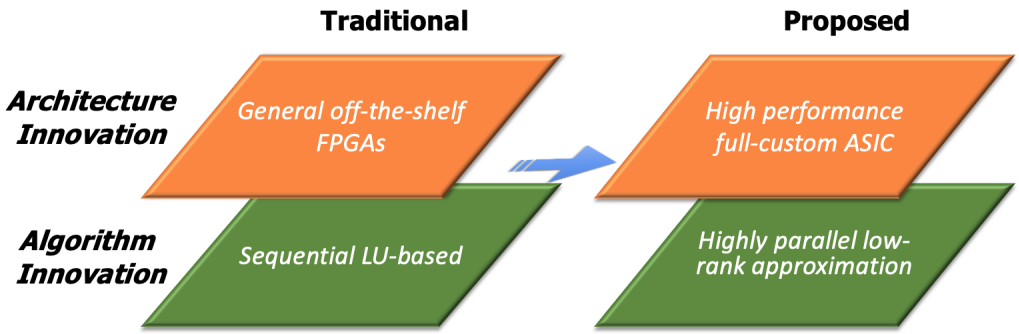


Figure 1.2: Two layers of innovation of the proposed research.

The rest of the dissertation is organized as follows. Chapter 2 introduces background and knowledge of EMT included models for various power system components used in EMT simulation and nodal analysis flow for EMT simulation. Chapter 3 and 4 explain the algorithm innovation where chapter 3 presents a hierarchical low-rank approximation based network solver, and compares the performance with the traditional approach by large-scale system, chapter 4 presents a hierarchical inverse computation and modification approach, and compares the performance with

traditional approach by large-scale system. Chapter 5 focuses on the architecture innovation which gives a detailed VLSI design for the proposed approaches. Chapter 6 is the conclusion of the dissertation.

2. ELECTROMAGNETIC TRANSIENT SIMULATION

In the power system, many devices can be modeled by different styles depending on the simulation purposes. This chapter introduces the generator, control element, and load models used in EMT, and also presents a general analysis flow handling different modeling scenarios in EMT simulation.

2.1 Background

In power system, EMT is a powerful tool which could perform very high-speed and wide electromagnetic transients and associated insulation issues ranging from seconds to nanoseconds. It was first developed by Hermann Dommel at Bonneville Power Administration to replace the transient network analyzer in the late 1960s. Initially, EMT is primarily used for transmission line switching studies. As the power system becomes complicated and applications become diversely, to meet the requirement, EMT becomes more and more powerful and the improvement becomes extremely important. In the current state, EMT has developed to handle extremely variety of models of generators, transmission line, transformers and other elements in the power system, and solve any network which consists of interconnections of resistance, inductance, capacitance, single and π-circuits, distributed-parameter lines, and certain other elements. Due to the wide and fast dynamic simulation, the EMT is highly demanded in many designs and studies, e.g. the component rating design, insulation coordination, lightning overvoltage computation, HVDC converter control and operation, and the design of protection and control [4]. Besides that, EMT simulation is also required when studying new problems such as geomagnetically-induced currents [5]. The Fig. 2.1 shows multiple power system transients in terms of time range.

2.2 System Models Adopted in EMT Simulation

The power system is a stiff system which consists of varities of components such as generators, control elements of generators like exciters, the network which contains transmission lines, protective elements and loads. Each of them can be formulated into different models depends on

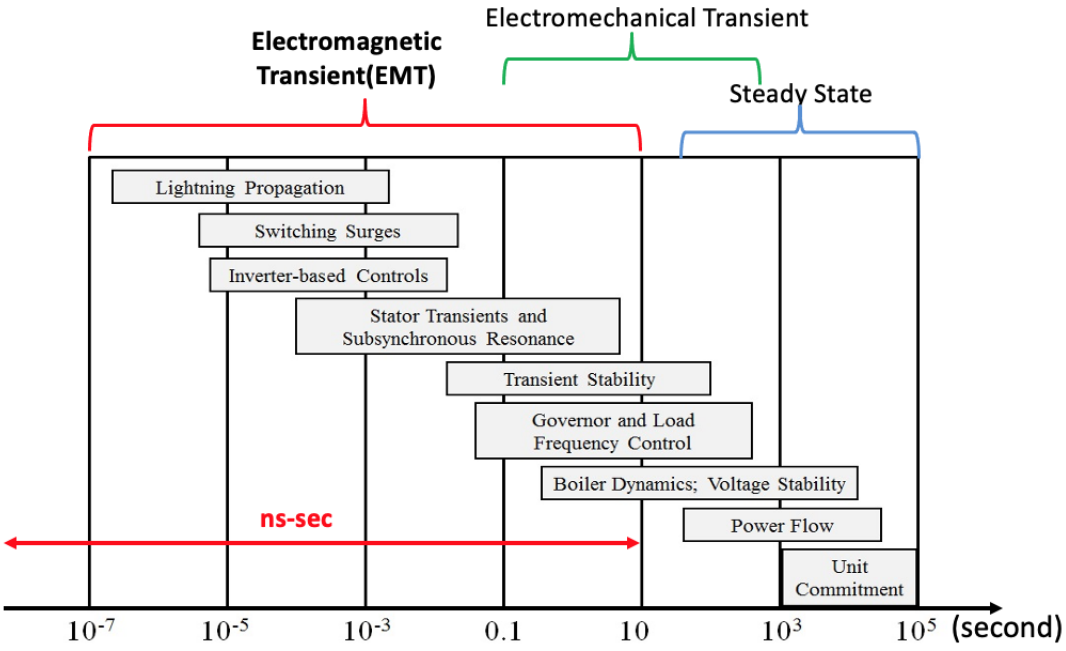


Figure 2.1: Time frame for different power system transients.

their needs. To get a better understanding of power system, let's start from a general structure with generators, transmission lines, exciters and loads.

The system of equations for EMT problem is shown in (2.1)

$$\begin{cases} \dot{x} = f(x, y) \\ 0 = g(x, y) \end{cases} \tag{2.1}$$

where x and y are state variables including the stator voltage in DQ0 axis, the stator current in DQ0 axis, the damper winding voltage in DQ0 axis, the damper winding current in DQ0 axis, the field voltage and current, the rotor speed of the synchronous machine, the mechanical and electrical torque of the synchronous machine.

It is a first order differential-algebraic equation (DAE) problem, where the generator model is a high order ODEs in the DQ0 axis connected to the network; the exciter is ODEs in the abc axis connected to the generators, and the order of ODEs depends on number of transfer functions; the network model is ODEs in the abc axis due to the existence of impedance; the loads are algebraic

equations in the abc axis connected to the network; and the interface which convert between DQ0 coordinate and abc coordinate between network and generator is the nonlinear algebraic equations which called park and inverse park transformation. The structure of power system is shown in 2.2.

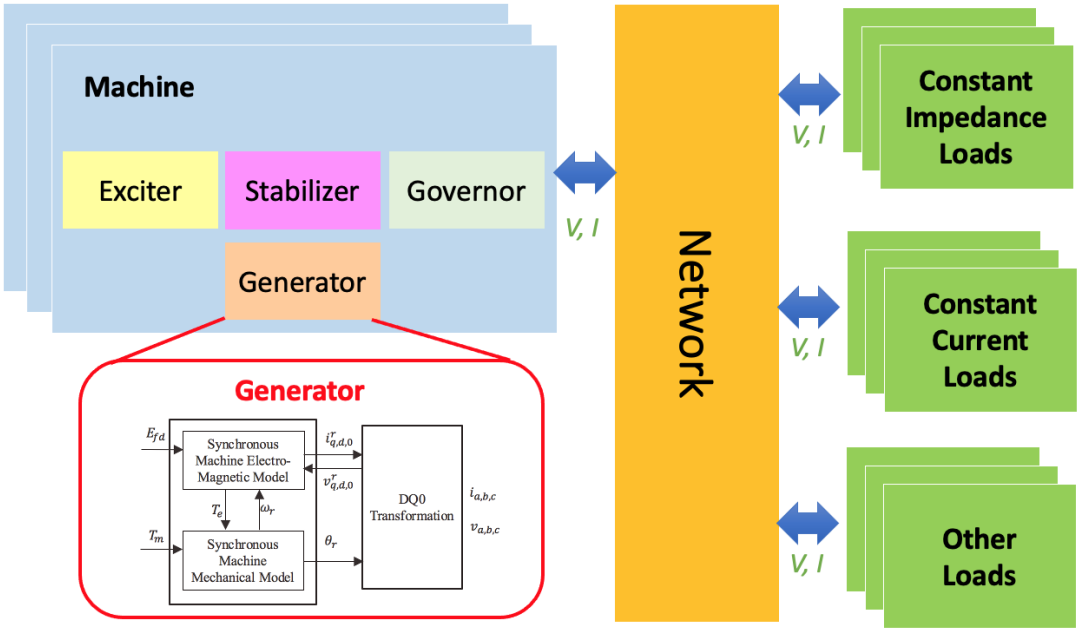


Figure 2.2: Power system structure.

For simplicity but without loss of generality, eight-order synchronous machine with two-orders exciter is represented as a voltage source behind an impedance, transmission lines are represented as a π model, transformers are represented by RL model, and loads are represented by constant impedance. The model details is described in Table 2.1. Note that, other kinds of models like higher order model of generators and control elements, Bergeron model of transmission line, and ZIP model of loads can be converted to the current models we adopted and mapped into the EMT simulation mentioned in the later sections.

Component	Current modeling	Note
Generator	8th-order DEs	Higher order model can be included when needed.
Exciter	2nd-order exciter	More detailed exciter models and other controllers, e.g., governor, stabilizer can be included when needed.
Transmission line	Π model	Bergeron model can be included when needed.
Transformer	RL model	Basic model and advanced model, i.e. UMEC, can be added when needed.
Load	Constant impedance	Other models, e.g., constant current, constant power and/or induction motor, can be included when needed.

Table 2.1: System model adopted in the dissertation.

2.2.1 Generator and Control Element Model

2.2.1.1 Generator

The equations of the 8th-order model of the synchronous generator is shown in 2.2, which is standard [22]:

$$\begin{aligned}
\begin{bmatrix} e_d \\ e_q \\ e_{fd} \\ v_{1d} \\ v_{1q} \\ v_{2q} \end{bmatrix} &= M_{gen}(\omega) \begin{bmatrix} i_d \\ i_q \\ i_{fd} \\ i_{1d} \\ i_{1q} \\ i_{2q} \end{bmatrix} + N_{gen} \begin{bmatrix} \dot{i}_d \\ \dot{i}_q \\ \dot{i}_{fd} \\ \dot{i}_{1d} \\ \dot{i}_{1q} \\ \dot{i}_{2q} \end{bmatrix}, \\
\dot{\omega} &= -\frac{DP}{2J}(\omega - \omega_b) + \frac{P}{2J}\omega(P_m - P_e), \\
P_e &= \frac{3P}{4}\left[L_{ad}(-i_d + i_{fd} + i_{1d})i_q - L_{aq}(-i_q + i_{1q} + i_{2q})i_d\right],
\end{aligned} \tag{2.2}$$

where e_d, e_q denote the stator voltage on the d axis and q axis, respectively; i_d, i_q denote the stator current on the d axis and q axis, respectively; v_{1d}, v_{1q}, v_{2q} denote the damper winding voltage on the d axis and q axis, respectively; i_{1d}, i_{1q}, i_{2q} denote the damper winding current on the d axis and q axis, respectively; e_{fd}, i_{fd} denote the field voltage and current, respectively; ω denotes the rotor

speed of the synchronous machine; ω_b denotes the base rotor speed of the synchronous machine; T_m, T_e denote the mechanical and electrical torque of the synchronous machine, respectively; J, D denote the inertia and damping of the synchronous machine; respectively; P denotes the number of poles of the synchronous machine; and L_{ad}, L_{aq} denote the stator magnetizing inductance on the d axis and q axis, respectively. Details of matrices $M_{gen}(\omega)$ and N_{gen} can be found in [22]. In this dissertation, the DQ0 transformation is performed under the rotor reference frame.

2.2.1.2 Control Element

Exciters are used to control the synchronous machine field voltage and current. There are three types of exciters based on different excitation power source: DC, AC, and static. The block diagram of BPA EA [23] is shown in Fig. 2.3. In the figure, V_T is the magnitude of bus terminal voltage, V_{REF}, V_{STB}, V_{MIN}, V_{MAX} and e_{fd} are the voltage reference set point, the power system stabilizer output signal which is zero in this case, upper voltage limitation, lower voltage limitation and generator field voltage respectively, T_R, T_A, T_{A1}, T_E and T_F are time constants, and S_E is the saturation function in terms of the generator field voltage E_{fd}.

A governor is a device used to measure and regulate the rotor angle speed of a synchronous machine. The block diagram of type BPA GG [23] which is one typical WSCC Type G governor model is shown in Fig. 2.4. In the figure, $\Delta\omega$ is the rotor speed difference, and P_M is the mechanical power applied to the synchronous machine.

A PSS is used to enhance damping of power system oscillations through excitation control. The block diagram of type PSS1A [23] which is one typical single-input stabilizer model is shown in Fig. 2.5.

In this dissertation, to provide necessary stationary rotating magnetic field and best simplify the system model, we adopt a two-order exciter as the only control element in the generator, described in Fig. 2.6.

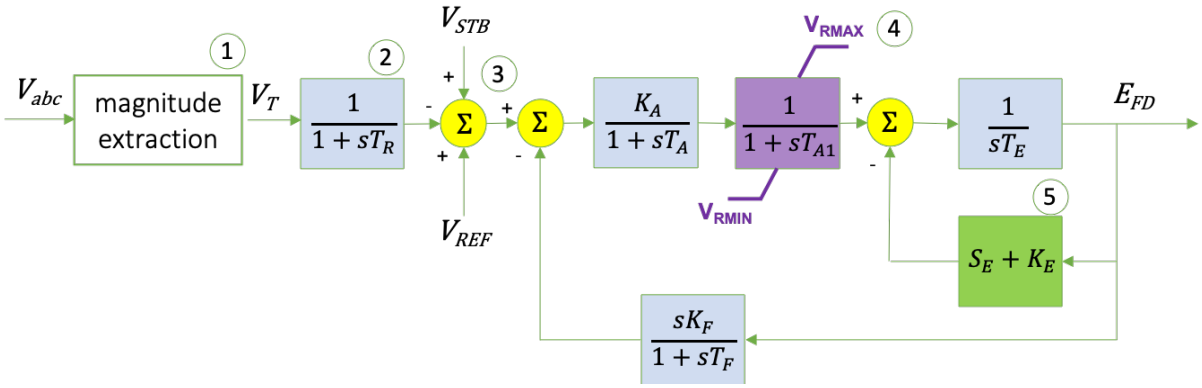


Figure 2.3: Exciter BPA EA — Continuously acting DC rotating excitation system model.

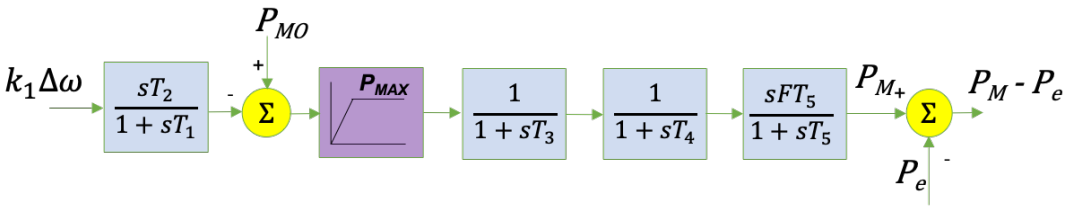


Figure 2.4: Governor BPA GG model.

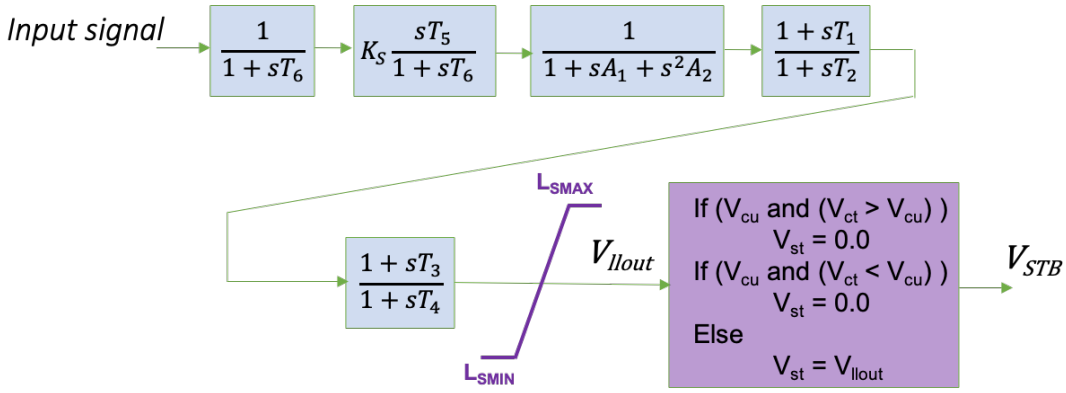


Figure 2.5: Stabilizer PSS1A — Single-input stabilizer model.

2.2.2 Transmission Line Model

Fig. 2.7 shows the three-phase Π section line model adopted in this dissertation. In Fig. 2.7, R_s, R_m denote the self and mutual resistances, respectively; L_s, L_m denote the self and mutual in-

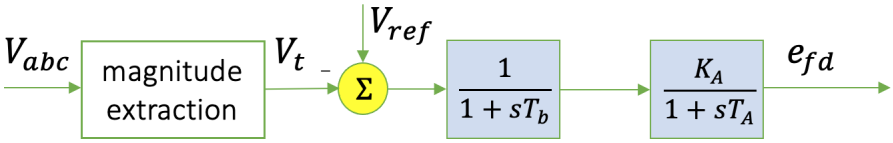


Figure 2.6: Second-order exciter model used in the dissertation.

ductance, respectively; and C_p, C_g denote the phase and ground capacitance, respectively. Details of the transmission line modeling can be found in [19, 22].

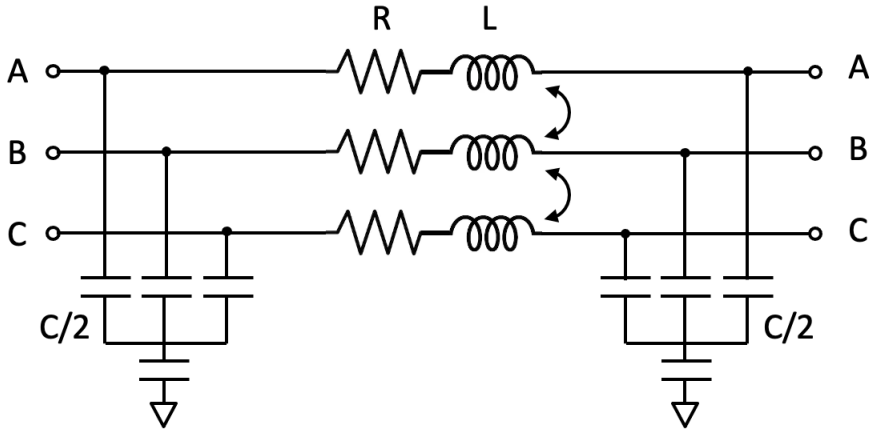


Figure 2.7: Three-phase Π model of the transmission line used in the dissertation.

2.3 Fault Model

In three-phase power system, the fault may involve one or more phases ground, or may occur short circuit between phases. Therefore, the fault events are divided into five types:

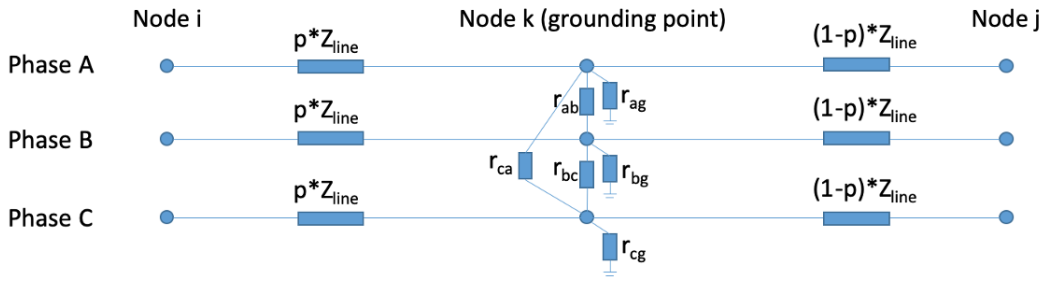
1. Single-phase-to-ground fault, which occurs when one conductor drops to the ground or comes in contact with the neutral conductor; This is the most common fault which takes more than 70-80%.

2. Two-phase-to-ground fault, which occurs when any two phases of the power circuit is short circuited to ground or neutral;

3. Three-phase-to-ground fault, which occurs when all three phases of the power circuit drop to the ground or neutral;

4. Phase-to-phase fault, which occurs when any two phases are connected together with low impedance;

5. Three phase fault, which occurs when all three phases are connected together with low impedance. This is the most severe fault type.



	Single-phase-to-ground (80%)			Two-phase-to-ground			Three-phase-to-ground	Phase-to-phase			Three phase
Phase A	x			x		x	x	x		x	x
Phase B		x		x	x		x	x	x		x
Phase C			x		x	x	x		x	x	x
Ground	x	x	x	x	x	x	x				

Figure 2.8: Eleven fault types in power system.

In general, these types share similar parameters and computation.

2.4 Nodal Analysis

In the current stage of EMT simulation, there are two main simulation methods. One is called state-space, the other is called nodal analysis(or EMT-type program, numerical integrator substitution, Dommel's method, nodal admittance method, companion circuit method, method of difference equation) [24]. Both of these methods discretize the ODEs to the algebraic equations and simulate transient phenomena at discrete intervals of time. The state space directly solves the system of ordinary differential equations and algebraic equations together, while the nodal analysis which is a non-iterative method, formulate all the devices into a universal Norton equivalent [6, 7] circuit after discretization using an implicit integration method. The comparison of these two approaches are shown in the Table 2.2.

	State space	Nodal analysis
Equations	$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$	$Gv = i + I_{his}$
Formulation	Network topological proper-tree and linearization	KCL and an implicit integration method
Disadvantages	Time-consuming re-formulation [25]; Pseudo-nonlinear (due to delay) [25]; Only for small systems (<50 states) [26]; No f-dependent device [27]	Not flexible to use variable stepsize.
Users	Simscape, Opal-RT/eMEGAsim	PSCAD/EMTDC, RTDS, ATP, EMTP-RV, Simscape, Opal-RT/eMEGAsim.

Table 2.2: Comparison between state space and nodal analysis.

Since the state-space method has time-consuming reformulation issue and not suitable for frequency devices, we adopt the nodal analysis approach to integrate all linear/nonlinear electrical elements, e.g., generator, transformers, lines and loads.

2.5 Network Equations

Consider about the model we mention in the Table 2.1, we discretized the loads and transformers as RLC circuit and combine into network with transmission line. After numerical discretization by an implicit integration method and a choice of the time step of nodal analysis, the network is formulated as a universal equations shown in (2.3).

$$G\mathbf{v}(t) = \mathbf{i}_{\text{in}}(t) + \mathbf{i}_{\text{his}}(t - \Delta t), \tag{2.3}$$

where G is the sparse network conductance matrix, $\mathbf{v}(t)$ is the nodal voltage vector which need to be solved, $\mathbf{i}_{\text{in}}(t)$ is the vector of current injections, and $\mathbf{i}_{\text{his}}(t - \Delta t)$ is a real number vector in each simulation step representing history state of system in each simulation step which is known. It depends on a few states of generators and control elements at t, which need to be predicted at first. Note that G also depends on the integration time step and the integration method. Consider about numerical stability, we use trapezoidal rule which is an implicit method to discretize the ODEs equations.

2.6 Simulation Flow

Fig. 2.9 drafts the EMT simulation flow using nodal analysis approach, which our method is based on. In the Fig. 2.9, the left loop is the time marching loop, while the right loop updates system states when fault occurs. In order to solve generators and network at the same time and maintain numerical stability, we discretize the ODEs equations by trapezoidal rule, then predict some variables to update network, after solving the network, new variables are used to correct the prediction one and update the left variables.

2.7 Conclusion

The dynamics of the grid evolve over a wide range of time-scales. Among the fastest and most difficult ones to handle is the propagation of electromagnetic transient dynamics that are of microseconds time-scale. State-of-the-art EMT simulators in the market are highly inadequately

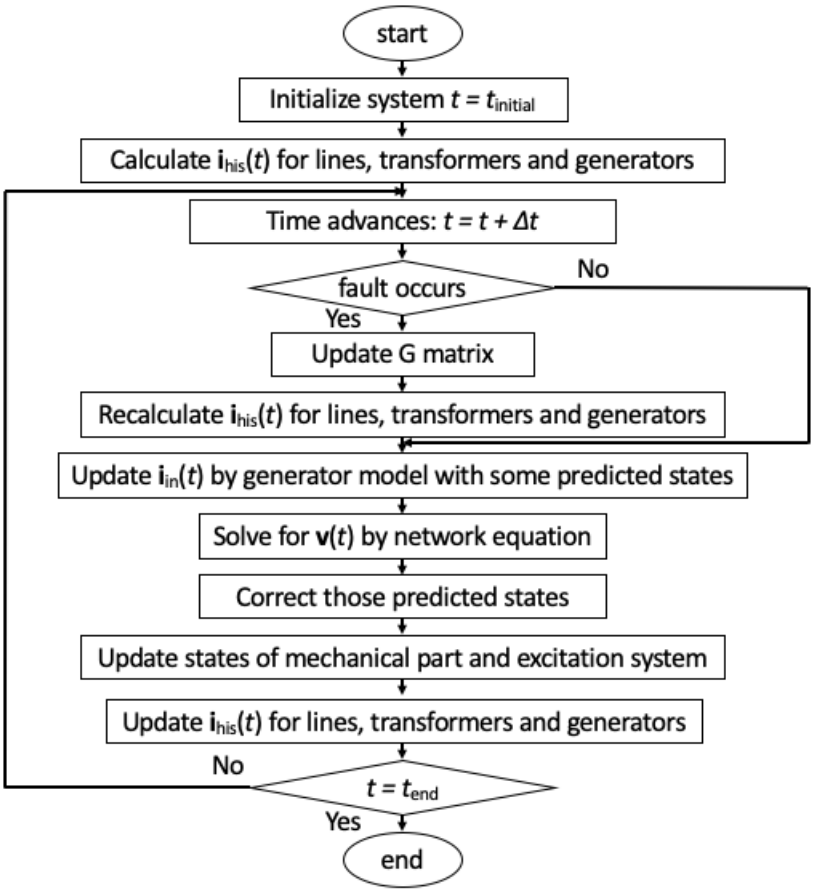


Figure 2.9: The EMT simulation flow using nodal analysis.

slow when simulating large-scale systems. The EMT simulator EMTP-RV, widely-used in the power industry, takes more than 80 hours to simulate 1-second of the dynamics of a distribution system with more than $80K$ nodes, and can therefore only be used offline. On the other hand, for real-time EMT simulation, the newest hardware by RTDS Technologies is based on IBM's POWER8 processor has a limit of around $4K$ buses [15]. Therefore, a fast than real-time EMT simulation is highly demanded in practice especially for modern complex power systems.

3. A HIERARCHICAL LOW-RANK APPROXIMATION BASED NETWORK SOLVER FOR EMT SIMULATION

In this chapter, we propose a novel approach to solve the network equations through a hierarchical low-rank approximation of the inverse of conductance matrix and fast matrix-vector multiplication. Overcome the traditional LU factorization-based approach's shortcoming, the proposed approach can implement in highly parallel. The performance of the proposed approach is verified by large-scale power systems.

3.1 Background

In modern EMT simulation,a commercial EMT simulator takes up to 80 hours to simulate one second of transient for an $87,000$-node system [17], with the majority of the time cost, $80-97\%$ [18][19], being taken by the network solution. Thus, the network solution is a bottleneck of the EMT simulation, and acceleration of the network solution is crucial for speeding up the overall large-scale EMT simulation. The network equation is a set of linear algebraic equations given by (2.3). The network equation has to be solved once at every time step, or multiple times when nonlinear models or control require an iterative solution. That causes huge computation burden in large-scale system. In the current stage, there are three general approaches to solve it.

1. dense inversion and matrix-vector products,
2. direct dense solver,
3. direct sparse solver.

Among these approaches, most real-time simulators adopt sparse LU factorization-based direct sparse solver to improve the efficient computation of large network equations in (2.3). Inspired by Gaussian Elimination, the sparse LU solver first performs the LU factorization (assume that a re-ordering technique has already been applied at the offline stage) as shown in (3.1) in the offline, where L and U are lower and upper triangular sparse matrices, respectively. Then in the online

stage, the voltage vector v is solved by forward and backward substitutions described in (3.2), where y is an intermediate vector.

$$G = LU, \tag{3.1}$$

$$Gv = i \Rightarrow \begin{cases} Ly = i, \\ Uv = y. \end{cases} \tag{3.2}$$

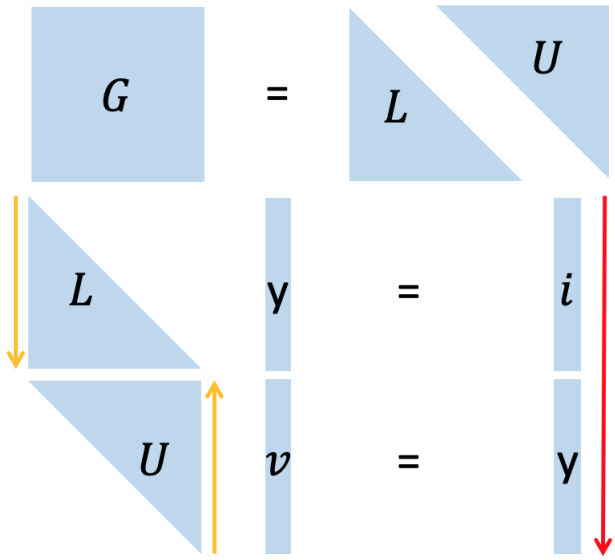


Figure 3.1: The LU factorization based solver.

The LU-based approach always outperforms the other two for systems with more than $2K$ dimensions. However, this approach is inherently sequential because of the forward and backward substitution. Therefore, direct sparse solver is not a good option for hardware implementation. We will propose a highly parallel approach to solve the linear system based our domain knowledge.

3.2 Network Equations

To exploit the parallelism, we rewrite the network equation as (3.3), and start from G inverse.

$$\begin{aligned} \mathbf{v}(t) &= G^{-1}\mathbf{i}(t), \\ \mathbf{i}(t) &= \mathbf{i}_{\text{in}}(t) + \mathbf{i}_{\text{his}}(t - \Delta t). \end{aligned} \tag{3.3}$$

Traditionally, solving the EMT network equation (2.3) consists of a pre-processing step where G is LU-factored, followed by repeated forward-backward substitutions at each time step. In contrast, the proposed direct inverse approach (3.3) first calculates the inverse of G in the offline stage in the offline, then do matrix-vector multiplication online.

3.3 Framework of the Hierarchical Low-rank Approximation Based Network Solver

As we mentioned, the bottleneck of (3.3) lies in fast solve the matrix-vector multiplication (mat-vec). Traditionally, mat-vec takes $O(N^2)$ time, however we will propose a novel approach that performs mat-vec in $O(N \log N)$ time. Just like traditional approaches, our proposed approach also requires a pre-processing step where G is inverted, the network is partitioned and the low-rank approximation on the partitioned network. To handle scenarios where G changes, e.g., topology change caused by loss of line, we could either prepare all involved G^{-1} matrices in advance and load them online when needed, or apply the fast modification approach introduced in Chapter 4. Since mat-vec is more efficient than forward-backward substitution, the overall performance improves when the number of time steps is high, as confirmed by experimental results. The diagram is in Fig. 3.2.

3.4 HOLDR Matrix

To illustrate the proposed approach clearly, let's define some notations first.

3.4.1 Block Matrix Notation

To better understanding the structure, we use the notation of Golub and Van Loan [28] to specify sub-matrices: If B is an $M \times N$ matrix, and $I = [i_1, i_2, ..., i_k]$ and $J = [j_1, j_2, ..., j_l]$ are index vectors, then $B(I, J)$ denotes the $k \times l$ matrix.

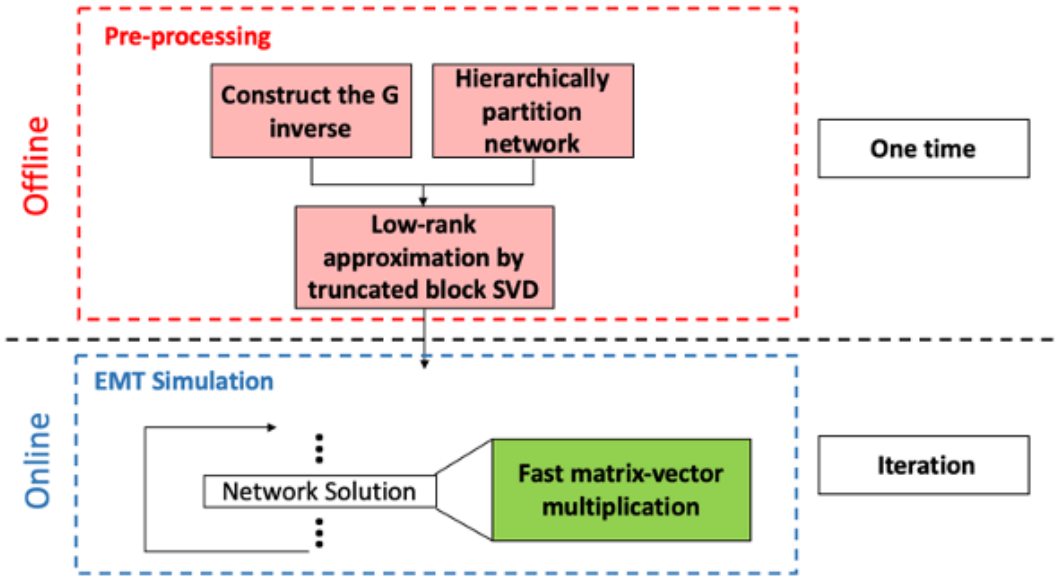


Figure 3.2: Framework of the hierarchical low-rank approximation based network solver.

$$B(I,J) = \begin{bmatrix} B(i_1,j_1) & B(i_1,j_2) & ... & B(i_1,j_l) \\ B(i_2,j_1) & B(i_2,j_2) & ... & B(i_2,j_l) \\ ... & ... & ... & ... \\ B(i_k,j_1) & B(i_k,j_2) & ... & B(i_k,j_l) \end{bmatrix}. \tag{3.4}$$

Let $B(I,:)$ be the matrix $B(I,[1,2,...,N])$ and define $B(:,J)$ analogously.

3.4.2 Matrix Partition Based on A Binary Partition Tree Structure

An $N \times N$ matrix B can be partitioned based on a partition of the index vector $I = [1,2,...,N]$ into a binary tree structure. Let I be the root of the tree and name the tree node as 1, e.g., $I_1 = I$. Then we split the root into two vectors I_2 and I_3 so that $I_1 = I_2 \cup I_3$. Node 2 and Node 3 are the two children of Node1. The full tree is then formed by continuing to subdivide any interval that holds more than some preset fixed number m of indices. We use the integer $l = 0,1,...,L$ to label the different levels, with 0 denoting the root level and L denoting the leaf level. In the leaf level, all nodes corresponding to vectors that never got split. The number of elements in each sub-index

vector I_k is n_k. These definitions are illustrated in Fig. 3.3.

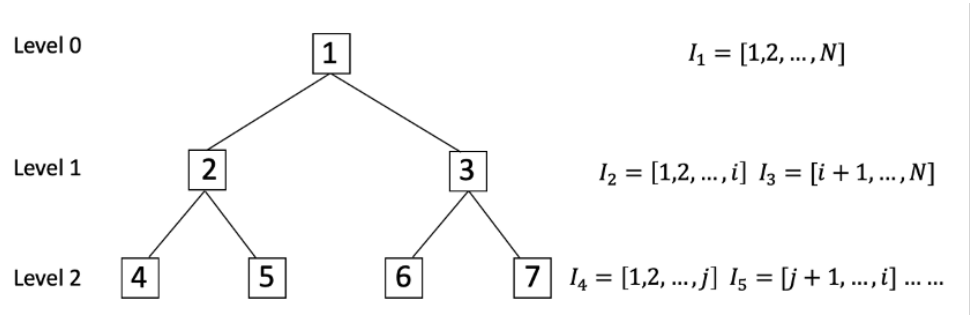


Figure 3.3: A binary tree structure of a matrix with index vector $I = [1, 2, ..., N]$.

With the hierarchical index vector partition, the matrix B correspondingly partitioned in a block format as Fig. 3.4, where D denotes the diagonal block matrix.

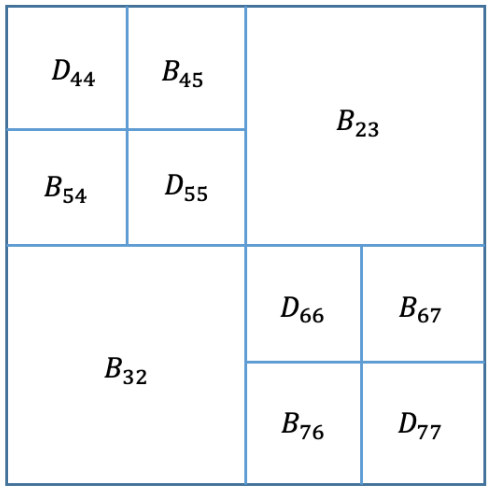


Figure 3.4: The partitioned matrix corresponding to the partition of Fig. 3.3.

3.4.3 HODLR Matrix Format

The HODLR which stands for the hierarchically off-diagonal low rank, is a kind of matrix with a property that the off-diagonal blocks should have low numerical rank. To be precise, given a

hierarchical partitioning of the index vector which create a binary tree, a computational tolerance ε, and a bound on the rank k, for any sibling pair (α, β), the corresponding block matrix $A(I_\alpha, I_\beta) = A_{\alpha\beta}$ has a rank lower than k which means the off-diagonal block can be represented by rank k approximation, shown as (3.5).

$$A_{\alpha\beta} \approx U_\alpha \widetilde{A_{\alpha\beta}} V_\beta, \tag{3.5}$$

where U_α and V_β are orthonormal matrices with size $n_\alpha \times k$ and $n_\beta \times k$ respectively. Suppose we require each leaf node to hold at most k points, where k often small, then it takes $O(kN \log N)$ storage to store all factors required to represent A, and a matrix-vector multiplication can be executed using $O(kN \log N)$ flops.

3.5 Low-rank Approximation for Network Equations

A low-rank approximation technique was previously used in solving PDEs for modeling process variations and for simulation of VLSI circuits [29], achieving significant speedup without sacrificing accuracy. Among the low-rank approximation approaches, Singular Value Decomposition (SVD) is the most suitable and powerful approach for EMT simulation since it is a numerically stable approach that can work on rectangular matrices, and is easy to implement.

3.5.1 Singular Value Decomposition

The SVD of a matrix $A_{m\times n}$ is given by (3.6).

$$A = U\Sigma V^*, \Sigma = \begin{bmatrix} \sigma_1 & 0 & ... & 0 \\ 0 & \sigma_2 & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \sigma_n \\ ... & ... & ... & ... \\ 0 & 0 & ... & 0 \end{bmatrix}, \tag{3.6}$$

where U is an $m \times m$ orthogonal matrix, Σ is an $m \times n$ diagonal matrix whose diagonal elements are non-zero singular values in decreasing order, e.g., $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0$, and V is an $n \times n$ orthogonal matrix.

The matrix format of SVD can be rewritten in a summation form as

$$A = \sum_{i=1}^{p} \sigma_i u_i v_i^*, \tag{3.7}$$

where σ_i is the i-th singular value of A, p is the rank of A and equals the total number of non-zero singular values, while u_i and v_i are the i-th left-singular vector and right-singular vector, respectively. Instead of summing over p terms in (3.7), we may take only the first r terms in (3.8) to approximate A, which is called the truncated SVD.

$$A \approx A_r = \sum_{i=1}^{r} \sigma_i u_i v_i^*. \tag{3.8}$$

We use the truncation error ε given by Eckart–Young–Mirsky theorem (3.9) as the indicator for the approximation error

$$\varepsilon = \|A - A_r\|_{\mathrm{F}} = \sqrt{\sigma_{r+1}^2 + \cdots + \sigma_p^2}, \tag{3.9}$$

where $\|.\|_{\mathrm{F}}$ is a Frobenius norm of a matrix.

The number of the left singular vectors u and right singular vector v of the truncated SVD are both r. In general, the less the value of r, the more we reduce the matrix and the max-vec time, but the higher the approximation error. Since ε is a good indicator for the approximation error, we select the r such that the corresponding ε is less than a pre-defined threshold ε_{th}.

3.5.2 Matrix-vector Multiplication using SVD

Traditional, the time complicity of (3.10) is $O(mn)$, where A is $m \times n$ matrix, both x and b are $n \times 1$ vector.

$$Ax = b. \tag{3.10}$$

Let's decompose the A by rank r truncated SVD, then compute the matrix-vector multiplication in the following three steps,

1. Compute the multiplication between V and x, which takes $O(rn)$ time complexity;

2. Compute the multiplication between the previous result and Σ, which takes $O(r^2)$ time complexity.

3. Compute the multiplication between the previous result and U, which takes $O(rm)$ time complexity.

Therefore, the time complexity of matrix-vector multiplication through SVD is $O(r(m+n+r))$, the process is described in Fig. 3.5. Since the r is relatively small compared with m and n, the truncated SVD permits matrix-vector multiplication great saving by reducing the FLOPs of matrix-vector product from $O(mn)$ to $O(r(m+n))$ which saves amount of computation especially when m and n are very large. Therefore, we could make use of SVD to speed up the network solution in EMT simulation.

3.5.3 Low-rank Approximation by Applying Truncated SVD

In EMT simulation, A in (3.10) represents the relation between input current variables and output voltage variables. When we write $A = U\Sigma V^*$, V represents a mapping from input current variables to an orthogonal current basis, Σ represents a diagonal matrix converting current basis to voltage basis, and U represents a mapping from orthogonal voltage basis back to output voltage variables.

Fig. 3.6 illustrates the key idea of using SVD to speed up the network solution. Fig. 3.6(a)-(c) shows the traditional approach to compute the impact of current in group S on voltage in group T, where buses in each group are closely connected but loosely connected between groups. Fig. 3.6(d)-(f) shows the proposed approach where the pairwise interaction matrix A, not necessarily

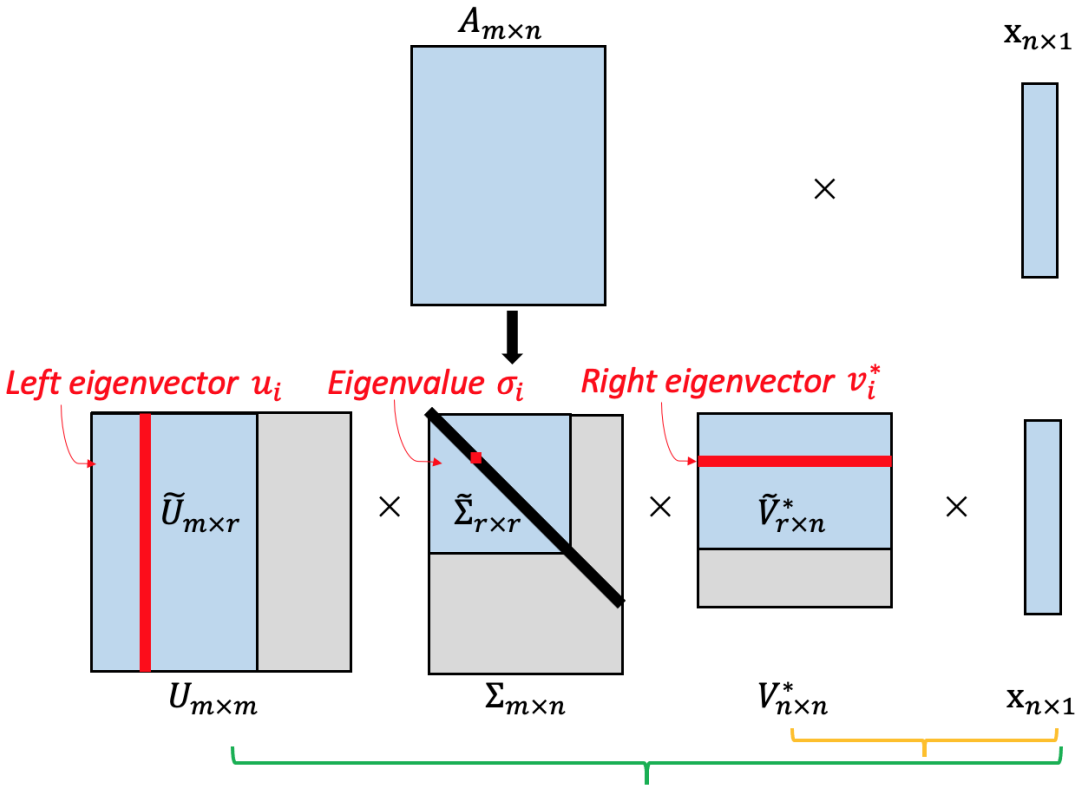


Figure 3.5: Matrix-vector multiplication using SVD.

square, represents any sub-matrix of G^{-1} which is approximated by a low-rank matrix A_r, thus significantly reducing the computation time.

The physical meanings for the truncated SVD in the EMT simulation are illustrated in Fig. 3.6. For the interaction between two groups of buses S and T is expressed by a matrix $A_{m\times n}$, where m and n are the number of buses in S and T, respectively. The truncated SVD maps both the m-dimension of S and n-dimension of T into an r-dimensional space formed by U and V, respectively. Therefore, the interaction between S and T can be simplified by the interaction between U and V over a linear transformation Σ. The impact of group S on T can further expressed as multiplication between U, V, Σ and input vector of S.

To calculate truncated SVD, we first compute the full SVD, and then use (3.8) to truncate the first r terms. For large dense matrices derived from the EMT, the truncated SVD can be computed by techniques like alternating-subspace methods [28], Monte Carlo, etc., without calculating the

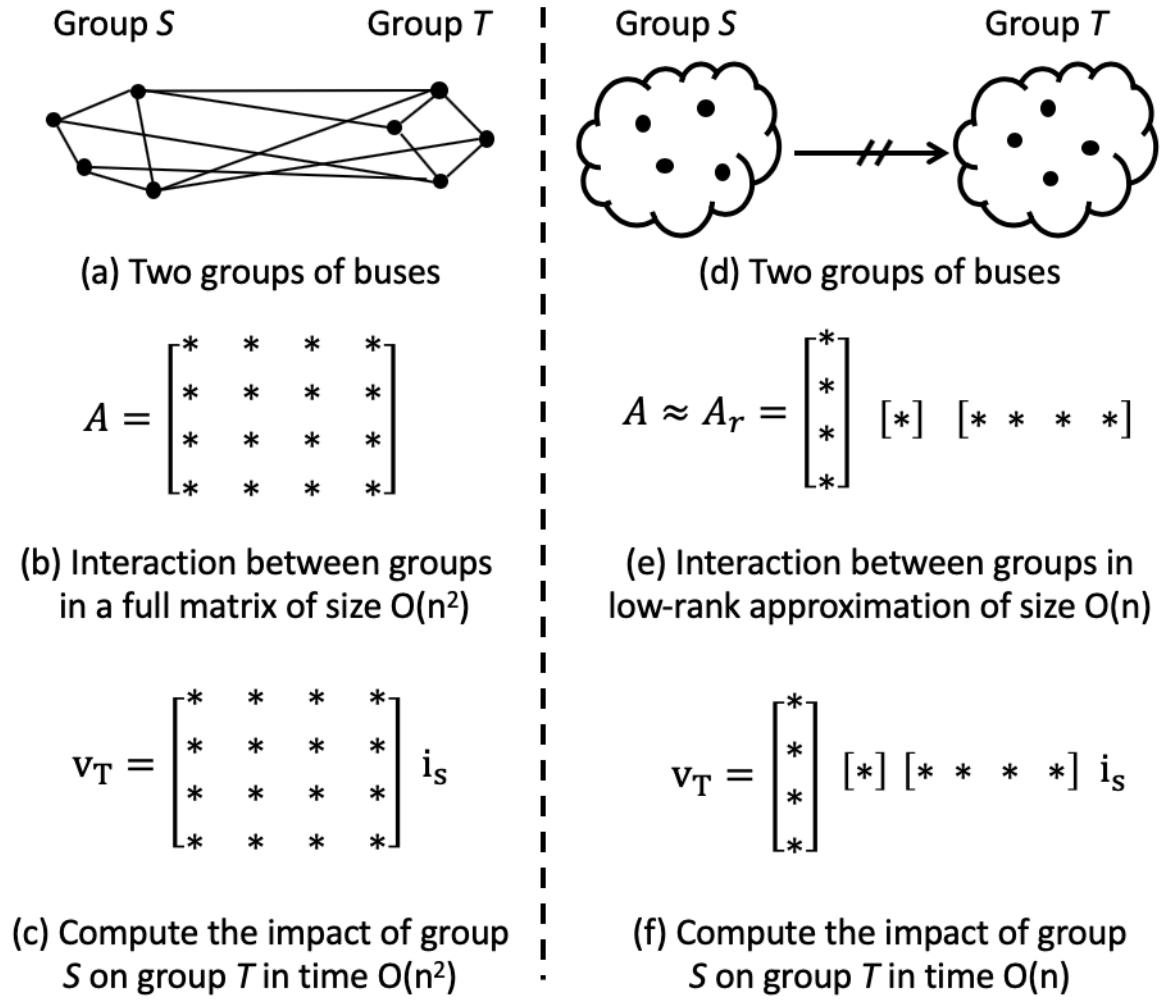


Figure 3.6: Using SVD to speed up the network solution between two groups.

full SVD.

3.6 Hierarchical Low-rank Approximation

The low-rank approximation is based on the observation that interactions between buses that are loosely connected, meaning the resistance between the buses are great or the connection between the buses is through multiple intermediate steps, can be approximated. However, the definition of closeness is relative: every bus has some buses that has close and some that are not. In order to systematically exploit the benefit of low-rank approximation, we propose the concept of hierarchically approximation, which is illustrated in Fig. 3.7(d)-(e). In comparison, Fig. 3.7(a)-(c) shows

the traditional approach to compute the impact of the network.

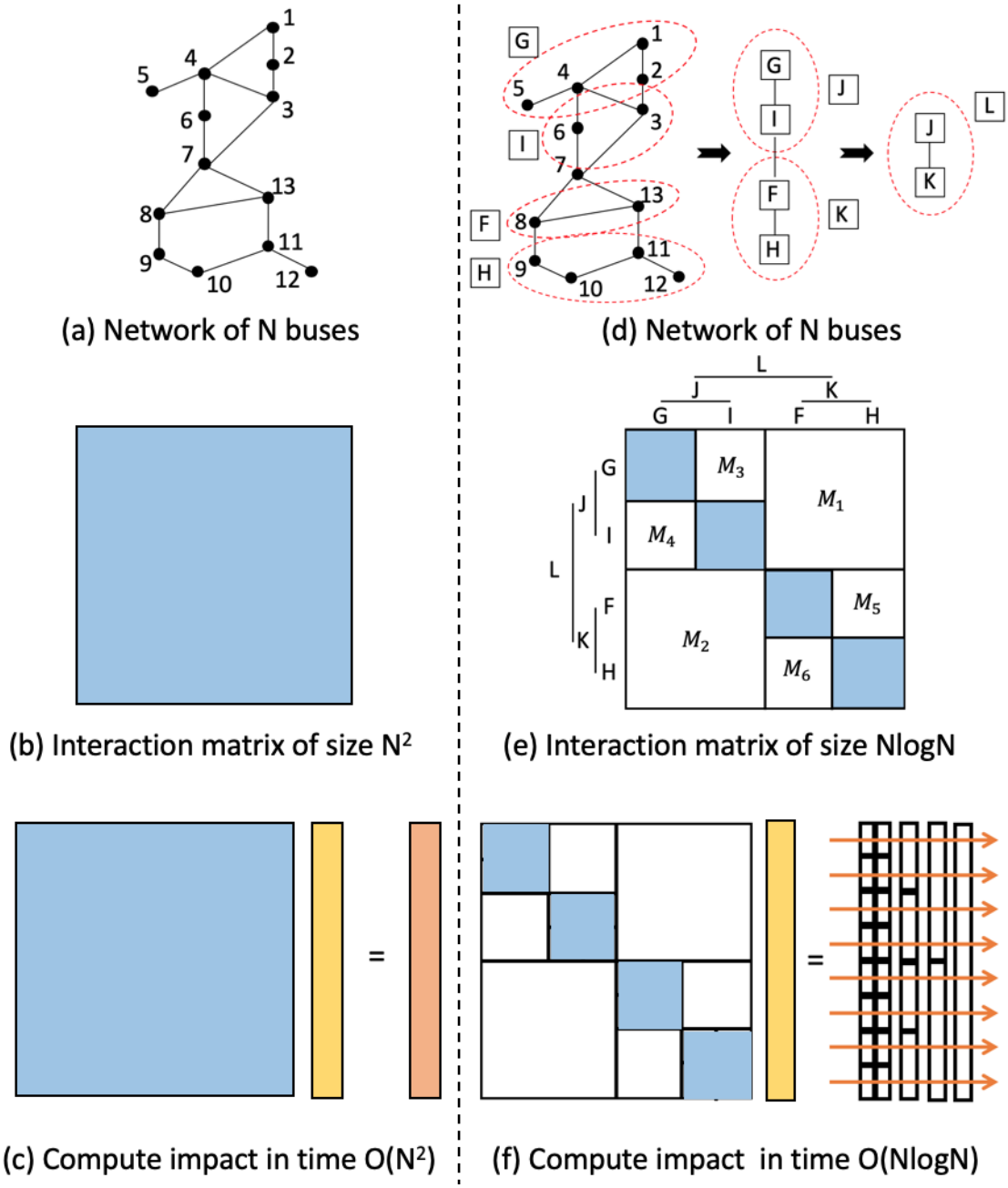


Figure 3.7: Using hierarchical low-rank approximation to speed up the network solution for whole network.

The interaction matrix among the buses is G^{-1}, which we want to approximate. However, it

is more difficult to tell which buses are close in G^{-1} than in G: if two buses share a transmission line in G, they are close. Therefore, we use G as the first criteria to select groups of buses that are may be partitioned. Nevertheless, we need a second criteria to determine if the interaction matrix between the two groups have a rank-r approximation, for a pre-defined threshold r_{th}. If so, then the interaction can be approximated. If not, consider the sub-networks.

Let's take an example in Fig. 3.7. Since group J consists of 7 buses and group K consists of 6 buses, the interaction between J and K is a 7×6 sub-matrix of G^{-1}, which is the M_1 sub-matrix in Fig. 3.7(e). The rows of M_1 correspond to buses in J and the columns correspond to buses in K. The algorithm first checks if interactions between J and K can be approximation with a matrix of rank no greater than r_{th}. In our example, the answer is yes, and M_1 is approximated by a truncated SVD. (If the answer were no, we would check interactions between G and F, G and H, I and F, and I and H.) The algorithm then recursively applies the same process to all sub-matrices, i.e., we check interactions between G and I, and between F and H. In general, if any connection has a low rank approximation, we save the approximation and stop the process for that branch. If not, we continue to its sub-networks. The process is continued until the entire network is either approximated or remains the same. Fig. 3.7(e) is the final hierarchical low-rank approximation, where white squares represent approximated matrices, and blue squares represent non-approximated matrices. Through this way, the G is represented in a HOLDR format.

The detailed algorithm is presented next. The inputs is the conductance matrix G^{-1}, which has been recursively partitioned into four sub-matrices. The details of the partition are introduced in Section 3.7. The outputs are the truncated SVDs. For each $m \times n$ matrix, the truncated SVD results are $U_{m\times r}, V_{n\times r}$ and $\Sigma_{r\times r}$, where $r \leq r_{\text{th}}$.

The benefit of the hierarchical low-rank approximation is illustrated in Fig. 3.7(f), where an $O(N \log N)$ time mat-vec can be performed. We will explain the mat-vec algorithm in more details in Section 3.8.

Algorithm 1: Hierarchical_low_rank_approx(G^{-1})

if $dimension(G^{-1}) \leq r_{\text{th}}$, **then**
  return G^{-1}
end
Calculate SVD of G^{-1}; Use ε_{th} to select the first r terms; Let the results be U, Σ, V^*;
if $r \leq r_{\text{th}}$ **then**
  return U, Σ, V^*
else
  Partition the row index of G^{-1} into R_1, R_2, and column index of G^{-1} into C_1, C_2.
  for all four sub-matrix G_{ij}^{-1}, *where* $i \in \{R_1, R_2\}, j \in \{C_1, C_2\}$, **do**
    Hierarchical_low_rank_approx(G_{ij}^{-1})
  end
end

3.7 Power Network Partition

The hierarchical low-rank approximation requires the inverse of conductance matrix G^{-1} be partitioned. The partition expresses G^{-1} as a hierarchy, where each sub-matrix, including G^{-1} itself, is further divided into 4 sub-matrices, until the rank of each sub-matrix is at most r_{th}.

Buses that are close to each other will have similar interactions with far way buses. Therefore, the partition should place buses that are close to each other in the same sub-matrix. Because the closeness of buses are reflected in the network, we will use a network graph $\mathbf{H} = (V, E)$ to guide the partition, where V is the set of vertices, representing the buses or a group of buses, and E is the set of edges, representing transmission lines or paths of transmission lines.

To reduce the computation time, the hierarchy should be balanced. Therefore, we propose a balanced bottom-up merging process, i.e., the buses are merged level by level from individual buses to the whole network. Two principles are used to guide the merging process. The first principle is to merge vertices with neighbors of highest conductance. This guarantees buses that are close will be in the same sub-matrix. The second principle is to merge each vertex only once during each level of the bottom-up process. This guarantees the partition is balanced.

The interaction between two sets of buses S_1 and S_2 can be measured by a weight function W,

which is essentially the average conductance between S_1 and S_2. Formally,

$$W(S_1, S_2) = \frac{\sum_{v_1 \in S_1, v_2 \in S_2} G_{v_i, v_j}}{max\{|S_1|, |S_2|\}}, \tag{3.11}$$

where G is the conductance matrix for the entire power system. Note that the denominator is $max\{|S_1|, |S_2|\}$ instead of $|S_1| + |S_2|$, because we want to assign higher weight for pairs of S_1, S_2 that are more balanced.

Following these principles, we perform the following actions at each level. First, we pick vertices of degree one and merge them, i.e., a vertex connected to only one neighbor is merged with its only neighbor. Then, for the rest of the vertices of degree greater than one, we repeatedly select pairs with the highest weight and merge each such pair, until no pair can be found. No vertex is merged twice at each level. If a vertex has only neighbors that were merged with other vertices, that vertex is left to be processed at the next level. After a level is completed, we go up to the next level, until the whole network is merged into one vertex. The detailed algorithm is presented below.

The graph partition algorithm thereby builds a binary tree data structure where the root represents the whole network, each internal node v'_k which has two children represents a group of buses v'_k merged from two groups v_i, v_j, and leaves represent individual buses in network.

An example of how the algorithm works is illustrated in Fig. 3.7(d). The vertices in the network graph are labeled with numbers, representing buses. The vertices labeled with uppercase letters represent groups of buses during the merging process. The process starts with merging individual buses into groups G, I, F and H. It then merges the pair G and I, and the pair F and H. Finally the pair J and K are merged into one group L, which contains all buses in the entire network. The buses or groups of buses in all merging steps are represented as nodes in a hierarchical partition tree, shown in Fig. 3.8, where leaves are buses, the root is the power grid. Apply Algorithm 1 starting from the root, the row and column indices of G^{-1} are derived from the partition hierarchically. The sub-matrices in Fig. 3.7(e), which are stored by U and V, are

Algorithm 2: Network_partition

Input: Graph: H $= (V, E)$, G^{-1}
Output: Tree root

$V_0 = V$; $E_0 = E$;
i = 0;
// bottom-up merging, level by level.
while $|V_i| > 1$ **do**
    (V_{i+1}, E_{i+1}) = Level_merge(V_i, E_i);
    i= i+1;
end
return V_i

Subroutine Level_merge(V, E)
    V' = ∅; E' = ∅;
    for *each* $v_i \in V$ *with only one neighbor* v_j **do**
        merge v_i and v_j into a new vertex v'_k;
        V' = V' ∪ {v'_k}; V = V – {v_i, v_j};
    end
    while *there are* $v_i, v_j \in V$ *and* $(v_i, v_j) \in E$ **do**
        pick $e = (v_i, v_j)$ with max weight;
        merge v_i and v_j into a new vertex v'_k;
        V' = V' ∪ {v'_k}; V = V – {v_i, v_j};
    end
    $V' = V' \cup V$; // add rest vertices
    $E' = (v'_i, v'_j)$,
    where v'_i, v'_j are merged from v_i, v_j respectively, and $(v_i, v_j) \in E$;
    return V', E'

represented as red dotted arrows in Fig 3.8.

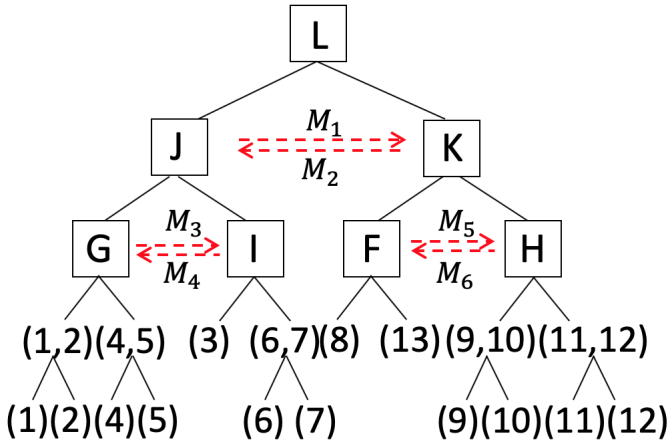


Figure 3.8: A hierarchical partition tree.

Theorem 1. *Given an N-bus power system network graph* $\mathbf{H} = (V, E)$ *where* V *is a set of vertices, representing buses, and* E *is a set of edges, representing transmission lines, Algorithm 2 produces a partition tree which has following properties,*

1. *The tree has one root which represents the whole network and N leaves which represents individual buses.*

2. *Each internal node represents a group of buses that are closely connected and has two children.*

3. *The height of the tree* h *is* $O(\log N)$.

Proof. According to the Algorithm 2, the proposed partition process starts from merging individual buses and ended up with merging into the whole network. During the merging, two groups of buses with high conductance are merged into one group. Therefore, the property 1), 2) and 3) holds.

According to property 1) and 3), the number of vertices in the bottom partition level is 2^h, thus $2^h = N$. Therefore, the height of the partition tree h is $O(\log N)$. □

Remark When there is a change in the topology, the network partition result may potentially be re-used, while the low-rank approximation has to be re-calculated.

3.8 Fast Network Solution

At each time step of EMT simulation, a network equation (3.3) needs to be solved, which is the most time consuming part of the entire EMT simulation. This section introduces a fast matrix-vector multiplication approach based on the hierarchical low-rank approximation of G^{-1} to solve the network equations. The analysis of time and error are also presented.

3.8.1 Fast Matrix-Vector Multiplication

Multiplication of G^{-1} with a vector $\mathbf{i}$ of bus current gives a vector $\mathbf{v}$ of bus voltage, which is the solution of the network equation. Traditionally, mat-vec is done in time $O(N^2)$, where N is the number of buses. In our algorithm, the mat-vec is done in time $O(N \log N)$, as shown in Fig. 3.7(f), which is called fast mat-vec. The fast mat-vec is made possible through the hierarchical low-rank approximation of matrix G^{-1} constructed in Section 3.6. The approximation represents matrix G^{-1} hierarchically in one of three forms:

1. as an explicit matrix of dimension at most r_{th}, which corresponds to the direct interaction among the buses.

2. as a low-rank approximation matrix of rank at most r_{th} in the form of U, V, Σ;

3. as a combination of its sub-matrices.

In the fast mat-vec, the output $\mathbf{v}$ is calculated hierarchically, level by level depending on the form of the sub-matrix. If the sub-matrix is represented in form 1), then compute the output $\mathbf{v}$ directly. If the sub-matrix is represented in form 2), then compute the output $\mathbf{v}$ by first multiplying $\Sigma(V\mathbf{i})$, followed by multiplying the previous product with U. If the sub-matrix is represented in form 3), the output $\mathbf{v}$ is computed by first multiplying $\mathbf{i}$ with children of the sub-matrix, followed by combing the results of the lower level. The algorithm is given below, where G_{ij}^{-1} is the sub-matrix, $\mathbf{i}_i$ and $\mathbf{v}_j$ are the corresponding input and output vectors, respectively.

Algorithm 3: Mat_Vec(G^{-1}, $\mathbf{i}$)

if G^{-1} *is stored as explicit matrix* **then**

$$\mathbf{v} = G^{-1}\mathbf{i}; \tag{3.12}$$

else if G^{-1} *is stored as* (U, V, Σ) **then**

$$\mathbf{v} = \sum_{i=1}^{r} u_i(\sigma_i v_i^* \mathbf{i}); \tag{3.13}$$

where r is the truncated rank of G^{-1}.

else

$\mathbf{v}_1$ = Mat_Vec(G_{11}^{-1}, $\mathbf{i}_1$) + Mat_Vec(G_{12}^{-1}, $\mathbf{i}_2$);

$\mathbf{v}_2$ = Mat_Vec(G_{21}^{-1}, $\mathbf{i}_1$) + Mat_Vec(G_{22}^{-1}, $\mathbf{i}_2$);

$\mathbf{v}$ = [$\mathbf{v}_1$; $\mathbf{v}_2$];

end

return $\mathbf{v}$

3.8.2 Time Complexity Analysis

Theorem 2. *Given a N-bus power system, the time complexity of solving network solution by Algorithm 3 is $O(N \log N)$.*

Proof. Let $T(N)$ be the time complexity of mat-vec in Algorithm 3, then

$$T(N) \leq \begin{cases} CN^2 & \text{if } N \leq r_{\text{th}} \\ CNr_{\text{th}} & \text{if rank } (G^{-1}) \leq r_{\text{th}} \\ 2T(\frac{N}{2}) + 2C(\frac{N}{2})r_{\text{th}} + N & otherwise, \end{cases}$$

where C is a constant and r_{th} is the user-defined rank threshold. Note that, in Algorithm 3, the time complexity of Mat_Vec($G_{11}^{-1}, \mathbf{i}_1$) and Mat_Vec($G_{22}^{-1}, \mathbf{i}_2$) are $T(N/2)$, while the time complexity of both Mat_Vec($G_{12}^{-1}, \mathbf{i}_2$) and Mat_Vec($G_{21}^{-1}$, $\mathbf{i}_1$) are $C(N/2)r_{\text{th}}$ due to the low-rank approximation. The term $T(N/2)$ for Mat_Vec(G_{11}^{-1}) and Mat_Vec(G_{22}^{-1}) is due to the balanced partition in Section 3.7.

$$
\begin{aligned}
T(N) &\leq 2T(N/2) + 2C(N/2)r_{\text{th}} + N \\
&= 2T(N/2) + (Cr_{\text{th}} + 1)N \\
&= 4T(N/4) + 2(Cr_{\text{th}} + 1)N/2 + (Cr_{\text{th}} + 1)N \\
&= \cdots \\
&= \sum_{i=1}^{\log N} (Cr_{\text{th}} + 1)N \\
&= O(N \log N).
\end{aligned}
$$

The sum from $i = 1$ to $\log N$ is due to the fact that the height of the partition tree is $\log N$, proved in Theorem 1. □

The time complexity bound in Theorem 2 may be reduce to $O(N)$ using an argument similar to Appel as follows:

1. Each node in the partition tree represents a set of buses, and each node in the tree interacts only with constant number of other nodes that are nearby and at the same level.

2. For buses that are far away, their interactions are through the nodes at higher levels.

3. For buses that are closer, their interactions are through the nodes at lower levels.

Therefore, the mat-vec time is proportional to the total number of interactions in the hierarchy, which equals the total number of nodes, which in turn is $O(N)$. The above argument is generally true for systems where the impact of any node decreases over a distance, which includes power networks. Indeed, our experiments indicate it is the case, although a rigorous proof is hard to derive due to exceptions in the man-made power network.

3.8.3 Parallelism of Fast Network Solution

The proposed approach partitions the whole network hierarchically into $O(\log N)$ levels, where N is the number of buses. At each level, the computation involves mat-vec multiplications of total time $O(Nr_{\text{th}})$, where r_{th} is the user-defined rank threshold. Therefore, with N processor units in parallel, the computation can be reduced to $O(r_{\text{th}})$. Since r_{th} is a constant, the proposed approach guarantees a network solution time of $O(\log N)$ using parallel techniques. This provides a significant advantage over LU based approaches.

3.8.4 Error Analysis

Theorem 3. *Let ε_{th} be the pre-defined error tolerance in the low rank approximation, such that*

$$\frac{\left\|\widetilde{M_i} - M_i\right\|_{\text{F}}}{\|M_i\|_{\text{F}}} \leq \varepsilon_{\text{th}}, \tag{3.14}$$

where M_i is the i-th sub-matrix and $\widetilde{M_i}$ is the approximation matrix of M_i. The relative error of the bus voltages in the EMT simulation is $o(\varepsilon_{\text{th}})$.

Proof. Denote the partition and approximation of G^{-1} as

$$G^{-1} = \begin{bmatrix} M_1 & M_2 \\ M_3 & M_4 \end{bmatrix} \text{ and } \widetilde{G}^{-1} = \begin{bmatrix} M_1 & \widetilde{M_2} \\ \widetilde{M_3} & M_4 \end{bmatrix}.$$

The relative error of bus voltage of the EMT simulation is bounded as follows,

$$\frac{\|\widetilde{\mathbf{v}} - \mathbf{v}\|_{\text{F}}}{\|\mathbf{v}\|_{\text{F}}} = \frac{\left\|\mathbf{i}(\widetilde{G}^{-1} - G^{-1})\right\|_{\text{F}}}{\|\mathbf{v}\|_{\text{F}}} \leq \frac{\|\mathbf{i}\|_{\text{F}} \left\|\widetilde{G}^{-1} - G^{-1}\right\|_{\text{F}}}{\|\mathbf{v}\|_{\text{F}}}. \tag{3.15}$$

Since $\|G^{-1}\|_{\text{F}}^2 = \sum_{i=1}^{n} \|M_i\|_{\text{F}}^2$, the squared error of G^{-1} is

$$\left\|\widetilde{G}^{-1} - G^{-1}\right\|_{\text{F}}^2 = \sum_{i=1}^{k} \left\|\widetilde{M_i} - M_i\right\|_{\text{F}}^2,$$

where k is the total number of approximation matrices. Then (3.15) is rewritten as

$$\frac{\|\widetilde{\mathbf{v}}-\mathbf{v}\|_{\mathrm{F}}}{\|\mathbf{v}\|_{\mathrm{F}}} \leq \frac{\|\mathbf{i}\|_{\mathrm{F}}}{\|\mathbf{v}\|_{\mathrm{F}}} \sqrt{\sum_{i=1}^{k} \left\|\widetilde{M_i}-M_i\right\|_{\mathrm{F}}^2}.$$

Recall the definition of ε_{th} in (3.14),

$$\begin{aligned} \frac{\|\widetilde{\mathbf{v}}-\mathbf{v}\|_{\mathrm{F}}}{\|\mathbf{v}\|_{\mathrm{F}}} &\leq \frac{\|\mathbf{i}\|_{\mathrm{F}}}{\|\mathbf{v}\|_{\mathrm{F}}} \sqrt{\sum_{i=1}^{k} \|M_i\|_{\mathrm{F}}^2 \varepsilon_{\text{th}}} \\ &\leq \frac{\|\mathbf{i}\|_{\mathrm{F}}}{\|\mathbf{v}\|_{\mathrm{F}}} \left\|G^{-1}\right\|_{\mathrm{F}} \varepsilon_{\text{th}}. \end{aligned} \tag{3.16}$$

Since the coefficient in front of ε_{th} in (3.16) only relies on the power system, the relative error of bus voltage of EMT simulation is proportionate to ε_{th}, which can be controlled through the appropriate choice of ε_{th}. □

3.9 Case Study

This section presents comprehensive case studies on two series of large power networks based on a 39-bus system and a 179-bus system (whose data and one-line diagrams can be found in [30][31]) to evaluate the accuracy, speed and scalability of the proposed **Hi**erarchical **L**ow-rank **ap**proximation approach, named "HiLap" hereafter. We compare our method against previous best method on both the accuracy and the speed. In addition, error - runtime tradeoff will also be presented. Details on the setup of tested cases and testing environment are provided below:

- **Generation of large cases**: To create large cases involving thousands of buses, we make multiple copies of a 39-bus system or a 179-bus system, and connect the copies in an array, shown in Fig. 3.9 and Fig. 3.10. This is the same approaches adopted [12][32]. To make the 39-bus system comparable with the 179-bus system, all the large cases of 39-bus system start from $39 \times 5 \times 1$ buses, whose G has a dimension starting from 585.

- **Accuracy test**: A large case based on the 179-bus system is created in EMTP-RV using Javascript, whose simulation results are used as reference. Both the error in system response

and the error in $|GV - I|$ are checked at each simulation step.

- **Speed test**: The CSPARSE package [33] is used as the sparse LU solver, named "SLU" hereafter. The SLU EMT simulation provides a reference result in all speed tests. In SLU, cs_lsolve and cs_usolve functions are used for the forward-backward substitution, and cs_amd are used for the natural ordering. The data structure used in SLU is Compressed Column Storage.

- **Fault simulation**: A three-phase fault is added at 10ms and cleared in 20ms. The post-fault period is simulated for 30ms, i.e., total simulation time = 60ms.

- **Simulation setting**: A 20ms simulation length and a 20μs time step are used in all tests.

- **Simulation implementation and testing environment**: All EMT simulations are programmed in C and conducted on an Intel Core i5 2.7GHz laptop, except the EMTP-RV simulation in Section 3.9.4.1 is performed on an Intel Core i7 3.47GHz desktop.

3.9.1 G and Approximated G Inverse

Fig. 3.11 depicts the sparsity of the original G matrix and the fill-ins created by the SLU approach based on a series of large systems created from 39-bus and 179-bus systems. In the series of 39-bus based system, the dimension of G goes from 585×585 to 5850×5850, the sparsity drops from 6% to 0.5%, and the number of fill-ins created by SLU grows linearly with system size up to 6×10^5. In the series of 179-bus based system, the dimension of G goes from 537×537 to 6444×6444, the sparsity drops from 6% to 0.5%, and the fill-ins created by SLU grows linearly with system size up to 1.4×10^6.

In our test cases, we set a rank threshold $r_{\text{th}} = 6$ and relative error tolerance $\varepsilon_{\text{th}} = 10^{-6}$, which are sufficient to maintain both the computational performance and the precision of final results. These choices are made through experiments and the fact that the interaction between any pair of three-phase buses is a rank-6 matrix.

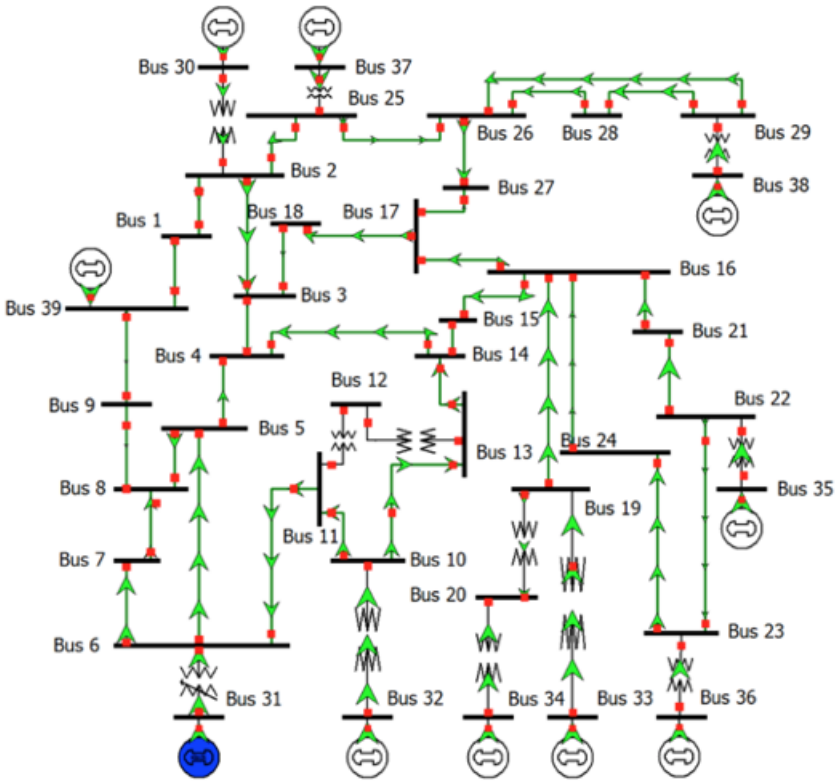


Figure 3.9: NE 39-bus system.

Fig. 3.12 depicts the approximated G^{-1} after applying Algorithms 1 and 2 for the for the 39-bus and 179-bus systems. The dimensions of G^{-1} are 117×117 and 537×537, respectively. In both cases, G^{-1} is divided into multiple sub-matrices with the rank of each sub-matrix no greater than 6. The approximated sub-matrices are high-dimensional and located in the off-diagonal, while the non-approximated sub-matrices are low-dimensional and located in the diagonal.

3.9.2 One-Step Network Solution

3.9.2.1 FLOPS Analysis

In order to measure the speed, we count the FLOPS involved in the solution of network equation for one time step. The FLOPS count includes total numbers of floating point addition and

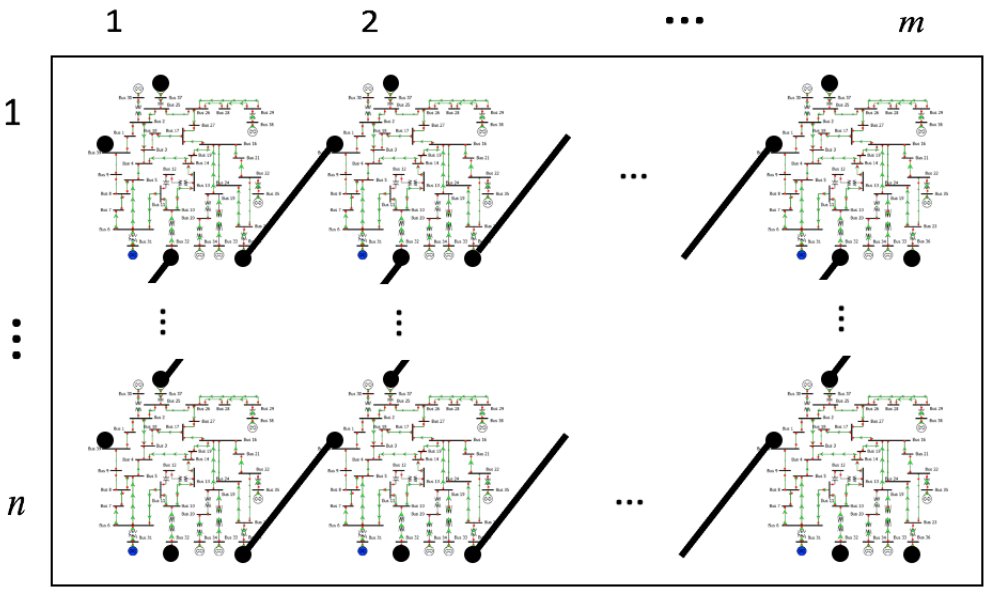


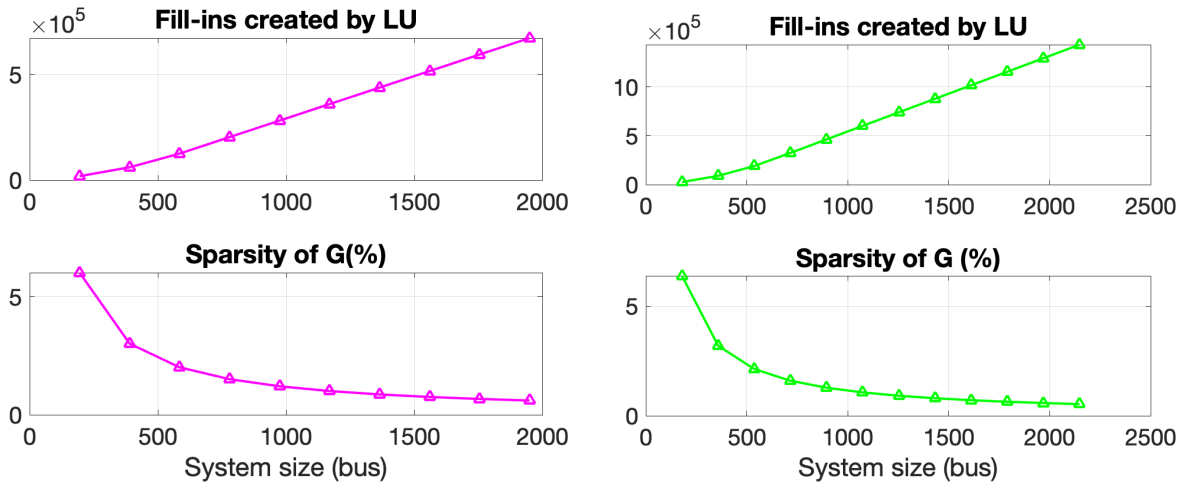
Figure 3.10: Large-scale power system based on NE 39-bus system.

multiplication operations per step. The results are shown in Fig. 3.13. In the series of 39-bus based systems with up to 1950 three-phase buses, the maximum FLOPS of SLU is 2.48×10^6, while the maximum FLOPS of HiLap is 1.36×10^6, leading to a 45% reduction. In the series of 179-bus based systems with up to 2148 three-phase buses, the maximum FLOPS of SLU is 5.56×10^6, while the maximum FLOPS of HiLap is only 1.43×10^6, leading to an 74% reduction.

3.9.2.2 Memory Analysis

Fig. 3.14 shows the memory requirements of SLU and HiLap. For SLU approach, only non-zero elements in L and U are taken into account, while for HiLap, the U, V, Σ, and the non-approximated matrices are taken into account.

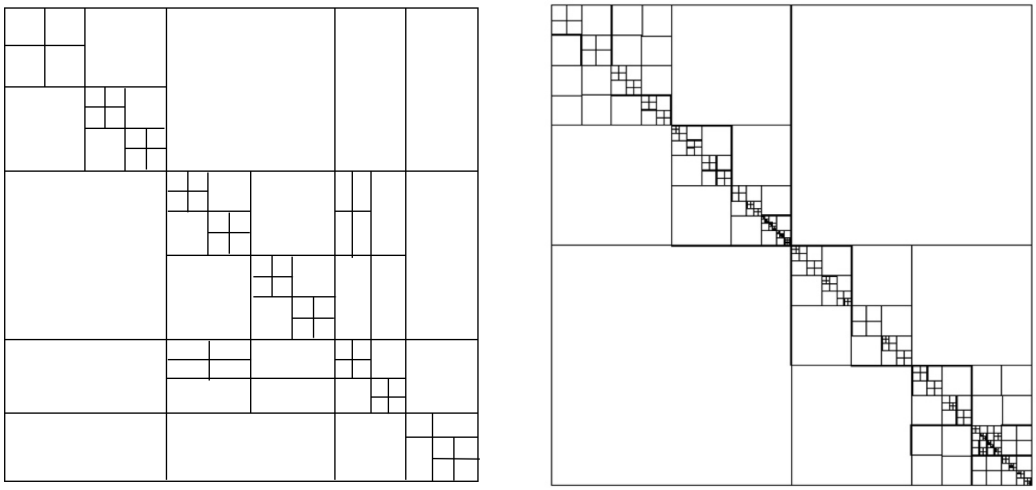
In the series of 39-bus based systems with up to 1950 three-phase buses, the maximum memory cost of SLU is 5.57MB, while it is 5.46MB for HiLap, a 2% reduction. In the series of 179-bus based system with up to 2148 three-phase buses, the maximum memory cost of SLU is 11.65MB, while it is only 5.73MB for HiLap, a 51% reduction.



(a) Series of 39-bus based systems

(b) Series of 179-bus based systems

Figure 3.11: Sparsity of G and number of fill-ins created by SLU.



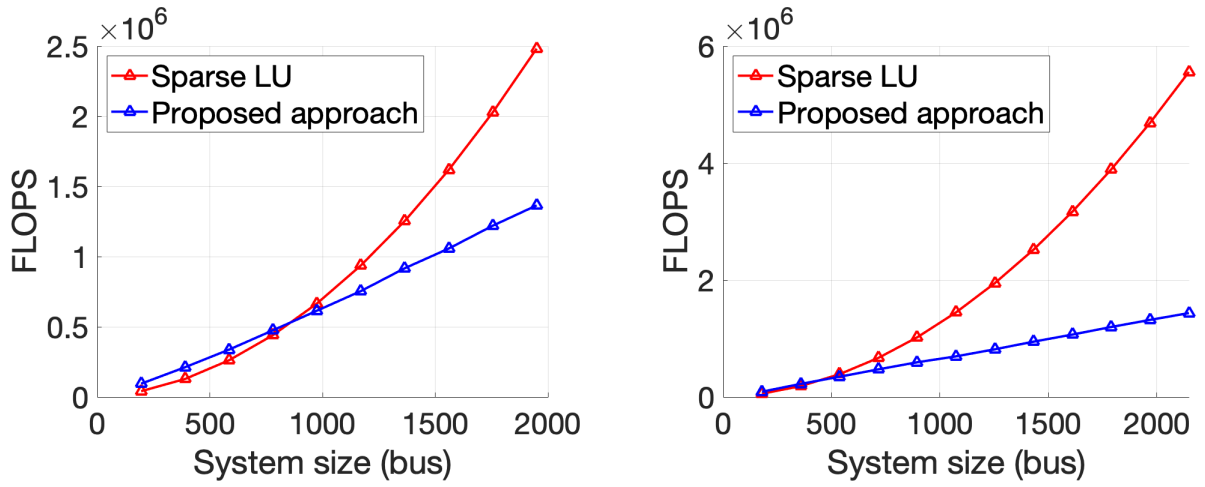
(a) 39-bus system

(b) 179-bus system

Figure 3.12: Hierarchical low-rank approximation of G^{-1}.

3.9.3 Runtime of Overall EMT Simulation

This subsection compares the runtime of network solution and overall EMT simulation. Fig. 3.15 shows the runtimes for series of 39-bus based systems. As shown in Fig. 3.15(a) and 3.15(b), the proposed HiLap outperforms SLU when the system size is larger than 975-bus. The maximum



(a) Series of 39-bus based systems

(b) Series of 179-bus based systems

Figure 3.13: FLOPS in one-step network solution.

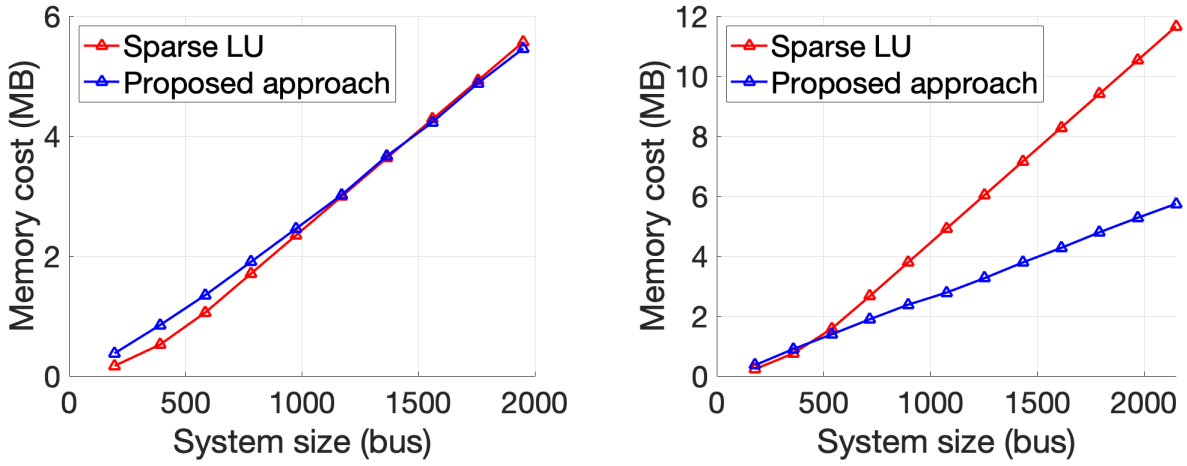
speedup ratios of network runtime is $1.7\times$ and the EMT runtime is $1.4\times$ with 1950 three-phase buses.

Fig. 3.16 shows the runtimes for series of 179-bus based systems. In the 2148-bus system, our approach is $2.8\times$ faster than SLU for the network solution part, and $1.85\times$ faster than SLU in the overall EMT simulation. Moreover, the runtime of solving for the network solution of our approach grows much slower than SLU, as Fig. 3.15 and Fig. 3.16 shows.

3.9.4 Accuracy Analysis

3.9.4.1 Benchmark with EMTP-RV

In this test, a 20ms simulation is conducted with a time step of 20μs. A 358-bus system created from the 179-bus system is simulated in EMTP-RV to provide a reference result. The same system is simulated by the proposed method and compared with this reference. It is observed that the HiLap result matches the reference well. Fig. 3.17 shows the three-phase bus voltages of bus 255. Fig. 3.18 shows the $|GV - I|$ error of bus 255 in this case, where the largest error turns out to be as small as 4.2×10^{-6} A.



(a) Series of 39-bus based systems

(b) Series of 179-bus based systems

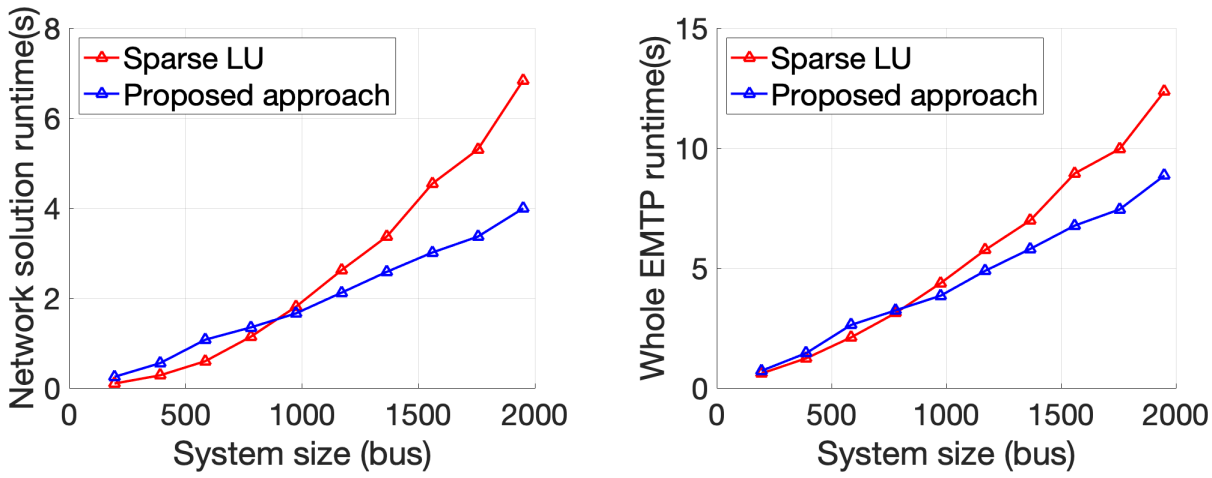
Figure 3.14: Memory requirements in one-step network solution.

3.9.4.2 Trade-off Between Error and Runtime

According to the analysis in Section 3.8, the user-specified error tolerance ε_{th} in the proposed approach has a significant impact on the simulation accuracy and runtime. Fig. 3.19 show that, as expected, ε_{th} is proportional to the maximum voltage error defined in (3.16), and inversely proportional to the runtime of network solution. In Fig. 3.19(a), there is a critical threshold of $\varepsilon_{\text{th}} = 10^{-7}$, since below this threshold, the voltage error is small enough and drops fast as ε_{th} decreases, while above this threshold, the voltage error increases slowly. For the runtime of the network solution as shown in Fig. 3.19(b), no significant improvement can be observed when $\varepsilon_{\text{th}} \geq 10^{-8}$. However, the runtime drops fast when $\varepsilon_{\text{th}} < 10^{-8}$.

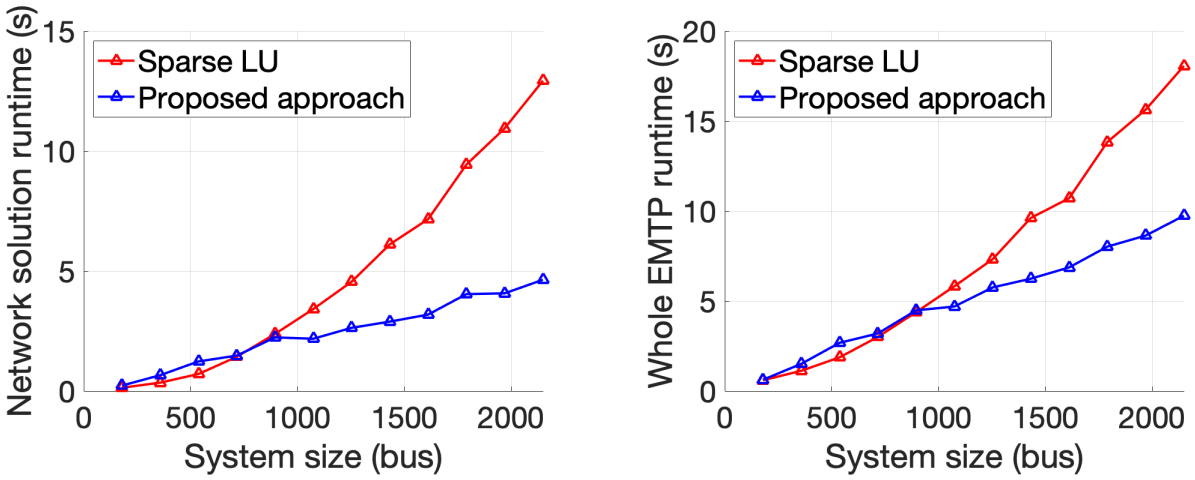
3.9.5 Fault Simulation

In this test, a three-phase fault is simulated with a time step of 20μs and a total simulation time length of 60ms. The fault is added at 10ms and lasts for 20ms. A 358-bus system created from the 179-bus system is simulated by SLU to provide a reference result. The same case is simulated by the proposed method and compared with the reference in Fig. 3.20(a) and Fig. 3.20(b), where the voltage of a bus near the fault is presented. It is observed that the result from HiLap matches that



(a) Network solution only (b) Overall EMT simulation

Figure 3.15: EMT runtimes test on serious of 39-bus systems.



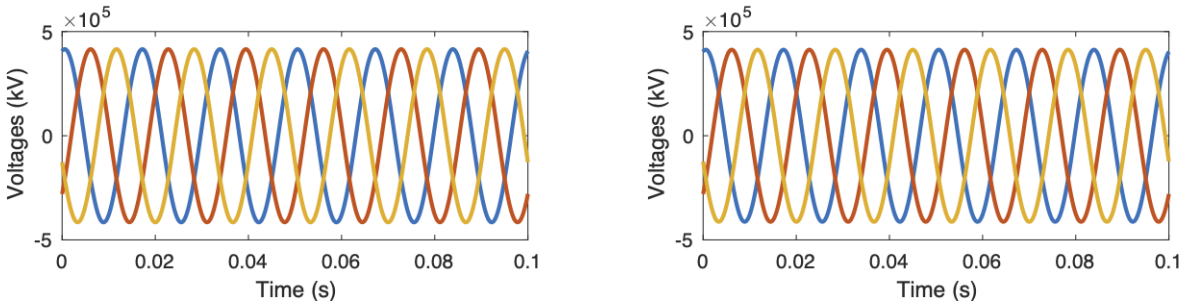
(a) Network solution only (b) Overall EMT simulation

Figure 3.16: EMT runtime test on series of 179-bus systems.

of the reference very well.

3.10 Discussions

The experiments show the proposed approach exhibits great speed improvement compared with SLU, and with high accuracy. As the size of the power network increases, the speed up



(a) Proposed method (b) EMTP-RV

Figure 3.17: Accuracy comparison to EMTP-RV.

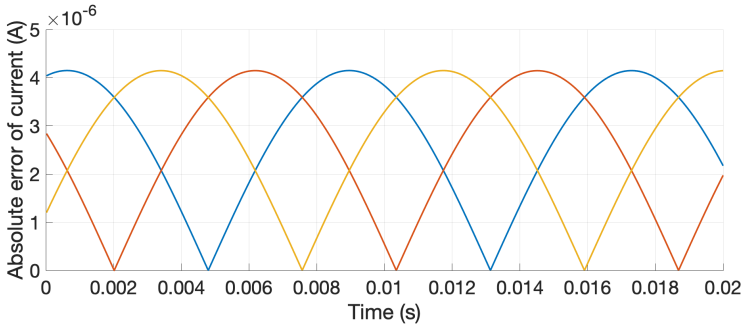
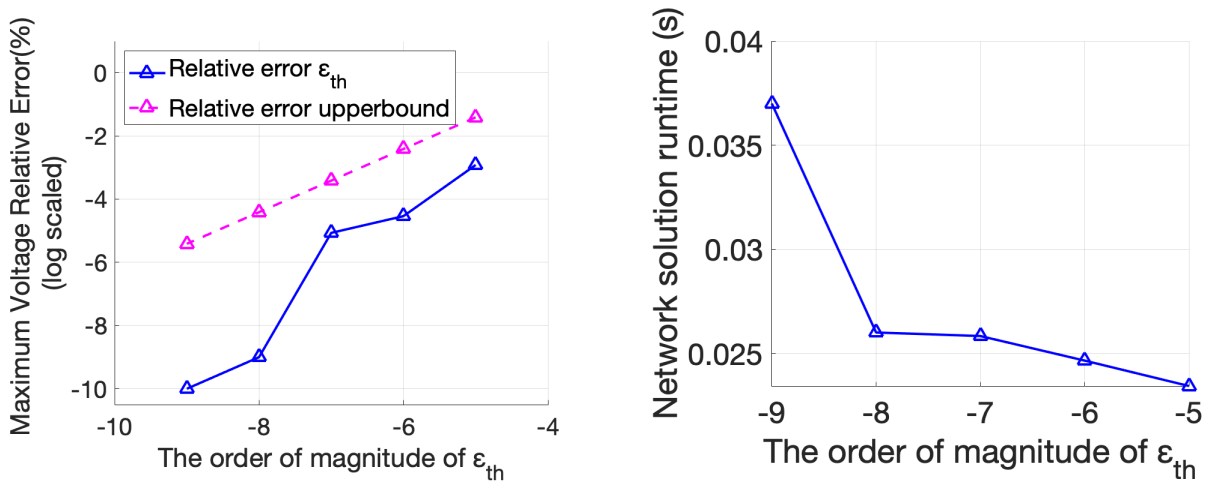


Figure 3.18: Approximation error in $|GV - I|$.

also increases. When decreasing the user-defined error bound ε_{th}, our simulation error decreases accordingly, while the run time increases slightly. Therefore, our proposed approach scales much better than previous approaches.

The experiments show that the topology of the power network has an impact on the performance of the proposed approach. To explain the difference in performance between the two series of systems, we compare a number of graph theoretic parameters, including number of edges, average degree, maximum degree, diameter, and average distance. We found the average distance between buses is a key indicator determining how tightly the buses are coupled. The average distance for the 39-bus system is 6, while for the 179-bus system is 14. As a result, the former has more non-approximated sub-matrics, while the latter has more approximated sub-matrics.

The time of the off-line step to compute G^{-1} and the low rank approximation in the experiment



(a) Relative error v.s. ε_{th}

(b) Network runtime v.s. ε_{th}

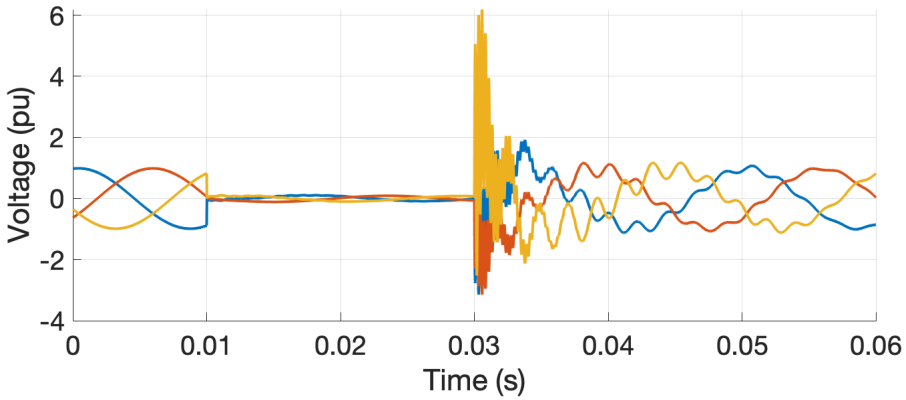
Figure 3.19: Impact of error tolerance on accuracy and speed.

is not counted, so is the LU factorization for the previous SLU approach. Since an EMT simulation needs to solve the network equation for hundreds of thousand of time steps, the time for the off-time step is amortized over all time steps.

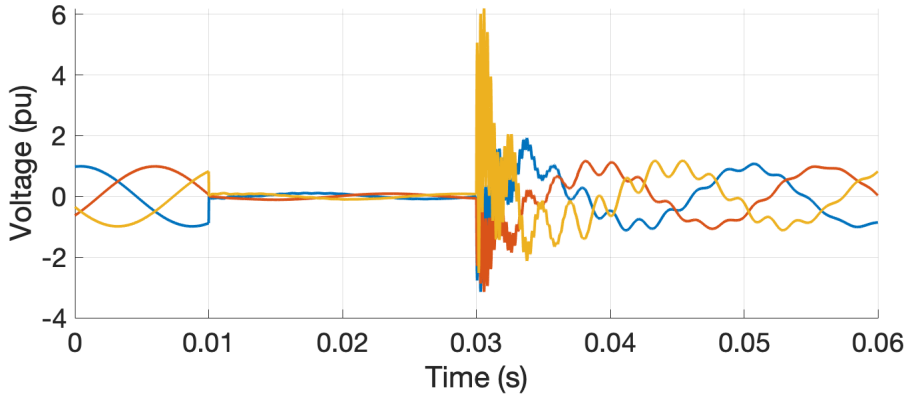
3.11 Conclusion

This chapter proposes algorithms to accelerate the EMT simulation in a new direction through a hierarchical low-rank approximation of the power network. Analysis of the performance and accuracy are presented. Case studies are performed on small- and large-scale systems. It is observed that the proposed approach can speed up the network solution by up to 2.8 times compared to the sparse LU factorization based direct solver and shows great scalability.

Compared with SLU which is inherently sequential in natural and not amenable to parallelization, the proposed approach has a hierarchical structure and thus highly parallelizable, using methods similar to [34]. Such parallelization may provide further speedup by high-performance hardware such as multi-core CPUs, GPUs, FPGAs or an ASIC.



(a) Proposed method



(b) SLU

Figure 3.20: Comparison of proposed method and SLU in a fault simulation.

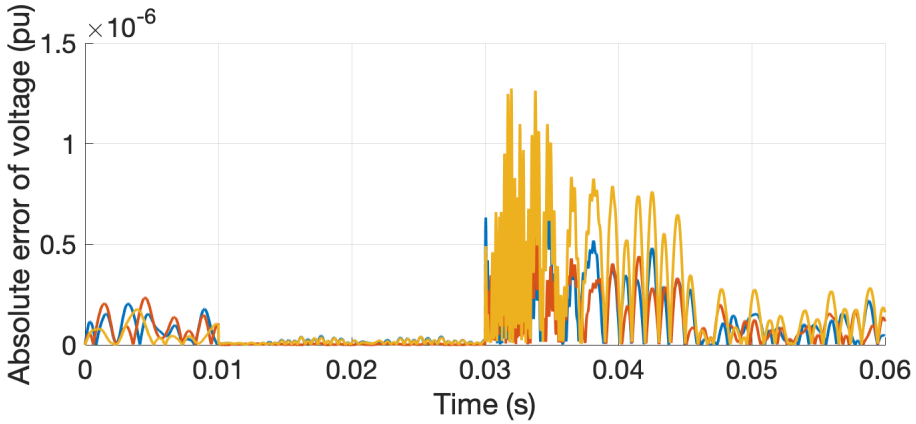


Figure 3.21: Voltage absolute error of proposed approach for fault simulation.

4. AN EFFICIENT NETWORK SOLVER FOR DYNAMIC SIMULATION BASED ON HIERARCHICAL INVERSE COMPUTATION AND MODIFICATION

In power system dynamic simulation, up to 90% of the computational time is devoted to solving the network equations, i.e., a set of linear equations. Traditional approaches are based on sparse LU factorization, which is inherently sequential. Therefore, we propose an inverse-based network solution that using hierarchical low-rank approximation and fast matrix-vector multiplication to speed up the network equations. However, the inverse-based approach requires significant time and memory to calculate and store the inverse of the large conductance matrix G in the pre-processing part. Also, this approach is not sufficient to modify the large conductance matrix G in the simulation when network changes by some faults, e.g., loss a line. Therefore, this chapter targets at fast computed and modified the impedance matrix caused by topological changed during the simulation. This chapter introduces a hierarchical method for computing and storage the approximate inverse of the conductance matrix. The proposed method can also efficiently update the inverse by modifying only local sub-matrices to reflect changes in the network. Experiments on a series of simplified 179-bus Western Interconnection demonstrate the advantages of the proposed methods.

4.1 Background

Numerical simulation of power system transients has been utilized in many critical applications in modern power industry, e.g., system dynamic security assessment, device insulation design and validation of control strategy [35][8]. In practice, simulation speed is always a key aspect of simulation tools, especially for real-time applications in large-scale systems [36]. As the penetration level of inverter-based generations, such as wind and solar, continues to increase in the upcoming decade, the complexity and dimension of the underlying systems will drastically increase along with more dynamic risks, e.g., sub- and super-synchronous oscillations. This poses a challenge to the inevitable need of fast dynamic simulations of large-scale systems [37].

Two major types of dynamic simulation in power industry are the electromechanical transient simulation, also called transient stability (TS) simulation, and the electromagnetic transient (EMT) simulation. In both types of simulations, solving the network equations, i.e., the linear equations, is the most time-consuming step, taking up to 90% of the overall computational time [18][19][38]. The most popular approach is based on LU factorization [33][39], which is adopted by many commercial simulation tools. This approach does the sparse LU-factorization in the pre-processing stage, then use right-hand-solving to solve the network equations during the simulation. If some events leading to changing of G, e.g., topology change caused by loss of line, it either loads the L and U ,which is well-prepared in the beforehand for all N-1 fault cases, of the updating G or redo the LU factorization of G on the fly. Regarding to the first solution, preparing all L and U for the N-1 fault in the offline is very exhausted especially when N is quit large. What's more, some fault scenarios does not easy to predict which causes preparing L and U of G is not practical. As to the second solution, redo the LU factorization during the simulation is very time consuming which will slow down the speed of EMT. The most important issue of the traditional approach is that the LU factorization and solving part is sequential in nature, making it difficult to be applied to parallel computing platforms. Therefore, devising a parallelizable fast network solver is an intuitive, while very challenging, way to speed up dynamic simulations.

In contrast, the inverse-based approach exhibits great potential in speeding up the network solution by parallel matrix-vector multiplication. Fig. 4.1 compares the LU-based and inverse-based approaches where black texts are our previous effort [40][41]. In [40][41], we explored a fast direct inverse-based network solver, where the solving step outperforms the sparse LU based network solver by about 2 times for large networks with $>$1000 buses in both FLOPS and runtime, which even shows a promising performance when sequentially computed. However, it requires significant time and memory to calculate and store the inverse of the large conductance matrix G the pre-processing part. Also, this approach is not sufficient to modify the large conductance matrix G in the simulation when network changes by some faults, e.g., loss a line.

Traditional, the computational burden and memory requirement of computing inverse elements

		LU-based Approach	Inverse-based Approach
Pre-processing	Method	Sparse LU Factorization	Hierarchical Approximation of G Inversion
	Complexity	Medium	~~High~~ Improve to Medium
Iteration Update	Method	Sparse LU Re-factorization	Fast G^{-1} Modification
	Complexity	Medium	~~High~~ Improve to Medium
Network Solution	Method	Right-hand-solving	Fast Matrix-vector Multiplication
	Complexity	Medium, not parallelizable	Low, parallelizable

Figure 4.1: Comparison between LU-based and inverse-based approaches, where red texts are contributions of this chapter.

for a large sparse matrix G is prohibitive even using efficient algorithms with optimization.

To further develop the technique of parallelizable network solvers, we propose a fast and memory-efficient approximation method for calculating and store the inverse of the large admittance/conductance matrices established from power grids. The new method can also efficiently update the inverse by modifying only local entries and sub-matrices to reflect changes in network, e.g., loss of a line. The advantage of the proposed method is demonstrated on EMT simulation, similar results can be expected in TS simulation as well.

The key contributions of this chapter are as follows.

1. An efficient hierarchical method for calculating and store G^{-1}, which can be used for large-scale power system inverse based simulation;

2. A fast and memory-efficient G^{-1} modification approach to handle scenarios where G changes, which may be used for fault simulation.

4.2 Hierarchical Approximation of G Inverse

The G^{-1} matrix plays an important role in power system analysis like contingency analysis. Unlike G matrix, G^{-1} matrix cannot be formed directly from the network. Even though G^{-1} can be found via inversion of G, explicit inversion is very expensive. Thus, this section introduces a hierarchical approximation approach for computing G^{-1} with much lower time and storage requirements.

4.2.1 Approximation of G Inverse

Assuming the transmission network does not contain any non-linear device, then G and G^{-1} are symmetric with real entries. Express G and G^{-1} in a block matrix format according to a partition of network G into sub-network G_1 and G_2,

$$G = \begin{bmatrix} G_1 & H \\ H^T & G_2 \end{bmatrix} \text{ and } G^{-1} = \begin{bmatrix} A & M \\ M^T & B \end{bmatrix}, \tag{4.1}$$

where the sizes of G_1 is $m \times m$, G_2 is $n \times n$, H is $m \times n$, and $m/3$, $n/3$ are the number of buses represented by G_1 and G_2, and each bus has three phases.

To compute the approximate G^{-1}, named as $\widetilde{G}^{-1}$, we want to find a matrix making the off-diagonals of $G\widetilde{G}^{-1}$ equal to 0, and the diagonals close to the identity matrix. Therefore, we wish to restrict sub-matrices A and B as follows,

$$\widetilde{G}^{-1} = \begin{bmatrix} G_1^{-1} & M \\ M^T & G_2^{-1} \end{bmatrix}. \tag{4.2}$$

We then find M and M^T.

$$\begin{aligned} G \times \widetilde{G}^{-1} &= \begin{bmatrix} G_1 & H \\ H^T & G_2 \end{bmatrix} \begin{bmatrix} G_1^{-1} & M \\ M^T & G_2^{-1} \end{bmatrix} \\ &= \begin{bmatrix} I + MH^T & G_1^{-1}H + MG_2 \\ G_2^{-1}H^T + M^T G_1 & I + M^T H \end{bmatrix}. \end{aligned} \tag{4.3}$$

Forcing the off-diagonals to be 0, we get

$$\begin{cases} G_1^{-1}H + MG_2 = 0, \\ G_2^{-1}H^T + M^T G_1 = 0. \end{cases} \tag{4.4}$$

Thus,

$$M = -G_1^{-1} H G_2^{-1}. \tag{4.5}$$

Plug (4.5) into (4.2),

$$\widetilde{G}^{-1} = \begin{bmatrix} G_1^{-1} & -G_1^{-1}HG_2^{-1} \\ -G_2^{-1}H^T G_1^{-1} & G_2^{-1} \end{bmatrix}. \tag{4.6}$$

In power systems, matrix $H_{m\times n}$ in G is determined by the network, i.e., the transmission lines connecting two groups of buses s_1 and s_2, where s_1 contains m buses, and s_2 contains n buses. Suppose there are k transmission lines between s_1 and s_2, then $H = h_1 h_2^T$, where h_1 is an $m \times k$ matrix representing the interaction from s_1 to s_2, and h_2 is an $n \times k$ matrix representing the interaction from s_2 to s_1. In general, k is much less than m and n. We can rewrite M as follows.

$$M = -(G_1^{-1}h_1)(G_2^{-1}h_2)^T. \tag{4.7}$$

If we multiply the matrices in the order given by (4.7) instead of (4.5), the computation cost will be reduced to $km^2 + kn^2 + kmn$. In addition, minimizing k will further reduce the time.

4.2.2 Hierarchical Approximation of G Inverse

To further reduce the time to compute G^{-1}, we propose the hierarchical approximation, which is illustrated in Fig. 4.2.

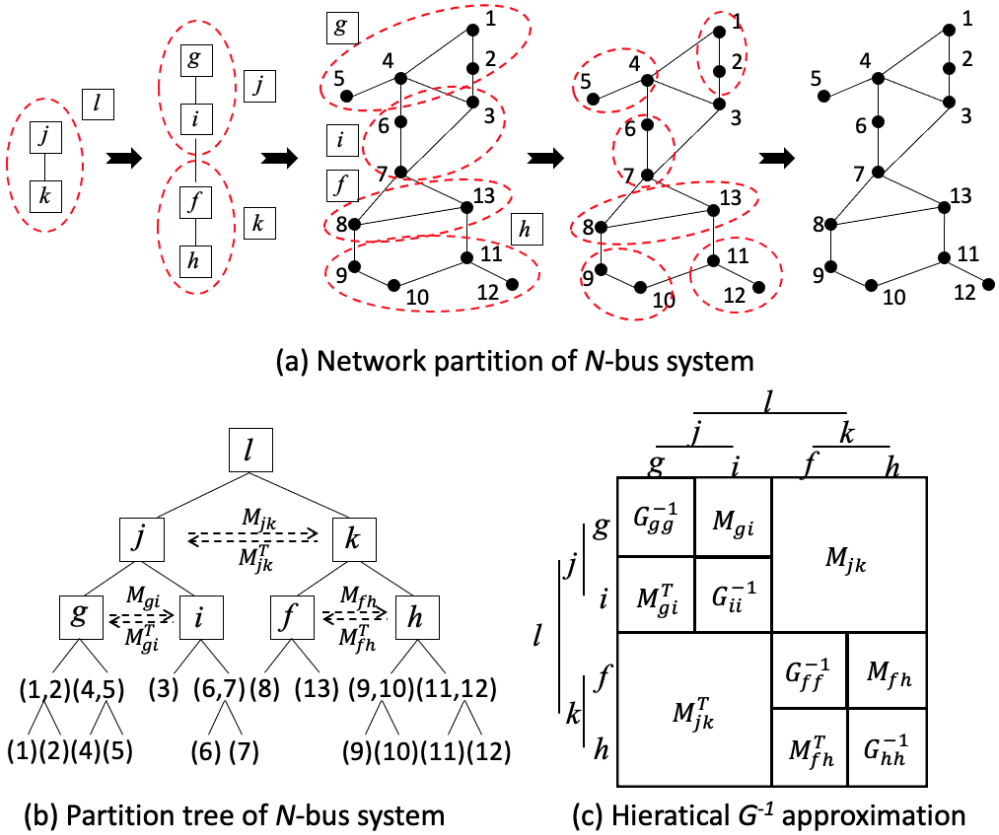


Figure 4.2: Hierarchical approximation for calculating G^{-1}.

The process is as described. First, we recursively partition the network into multiple sub-networks. In Fig. 4.2 (a), network l is first bi-partitioned into sub-networks j and k, then j and k are further bi-partitioned into sub-network g, i, and f, h, respectively, so on so forth until each sub-network contains only one bus. The partition thereby builds a binary tree where the root represents the whole network, the internal node represents a group of buses, and the leaf represents an individual bus, shown as Fig. 4.2 (b). Correspondingly, the network partition divides G^{-1} into a hierarchy, where each sub-matrix, including G^{-1} itself, is further divided into 4 sub-matrices, until

each sub-matrix is a single element. In Fig. 4.2 (c), group j consists of 7 buses and group k consists of 6 buses, thus the interaction within j is expressed by a 7×7 sub-matrix G_{jj}^{-1}, where the row and column indices correspond to buses in j; the interaction within k is expressed by a 6×6 sub-matrix G_{kk}^{-1}, where the row and column indices correspond to buses in k, and the interaction between j and k is expressed by a 7×6 sub-matrix M_{jk}, where the row indices correspond to buses in j, and the column indices correspond to buses in k. Based on the hierarchically partitioned G^{-1}, we apply the approximate inverse approach recursively, eventually construct the entire G^{-1} without any direct matrix inversion, i.e., G_{jj}^{-1} is constructed by G_{gg}^{-1}, G_{ii}^{-1} and M_{gi}, and M_{gi} is calculated by (4.7), where G_{gg}^{-1} is a 6×6 matrix, G_{ii}^{-1} is a 3×3 matrix, h_{gi} is a 6×3 matrix, and h_{ig} is a 3×3 matrix.

To improve the accuracy of the approximation, we pre-define a node threshold d_{th} to limit the size of the minimum group of buses. If a sub-network contains fewer than d_{th} buses, the partition stops. For the example in Fig. 4.3, we pre-define d_{th} equals 4, thus node g, i, f, and h are the minimal sub-network where the partition stops. To achieve high approximation accuracy and low computation cost, the network partition algorithm needs to minimize the edges between two sub-networks. In this chapter, we adopted the partition algorithm used in [41].

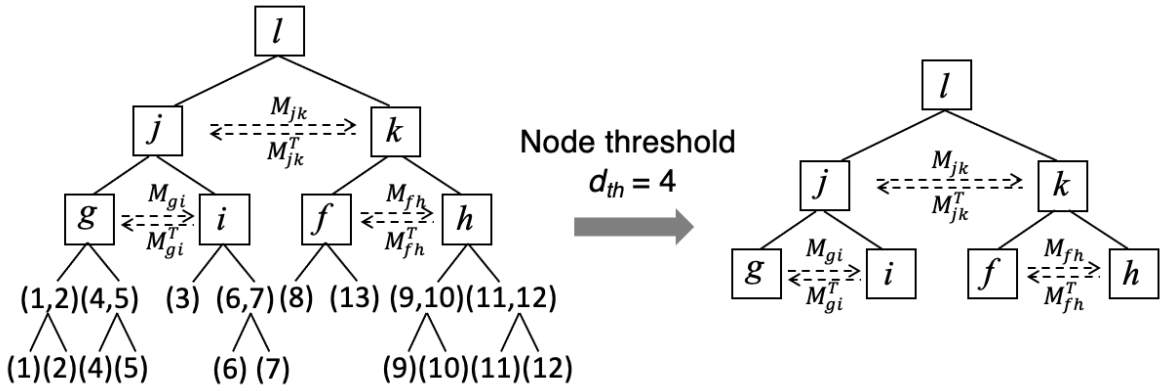


Figure 4.3: Partition tree of G^{-1} by node threshold d_{th} = 4.

The algorithm is presented below. The input is the root of the partition tree i, with its left child

being l and right child being r. Each tree node stores the bus indices of the sub-network. The outputs are M_{lr} and the approximation of G_{ii}^{-1}.

Algorithm 4: Hierarchical_Ginv(i)

if i has less than d_{th} buses **then**
  direct compute the G_{ii}^{-1};
else
  G_l^{-1} = Hierarchical_Ginv(l);
  G_r^{-1} = Hierarchical_Ginv(r);
  $H = G_{lr}$; Decompose $H = h_l * h_r^T$, where h_l is an $l \times k$ matrix, h_r is an $r \times k$ matrix, k is the number of edges between l and r;
  $M_{lr} = -(G_l^{-1}h_l)(G_r^{-1}h_r)^T$, $G_{ii}^{-1} = \begin{bmatrix} G_l^{-1} & M_{lr} \\ M_{lr}^T & G_r^{-1} \end{bmatrix}$;
end
return M_{lr}, G_{ii}^{-1}

The benefits of this approach are 1) Save computation of matrix inversion without direct invert G. In the algorithm, we only need to direct invert the diagonal matrices of each size bounded by node threshold d_{th}; 2) Reduce storage requirement. Instead of storing the entire dense matrix G^{-1}, this approach only requires to store diagonal matrices.

4.3 Fast G Inverse Modification

This section introduces a fast G^{-1} modification approach based on Algorithm 4 to handle network changes.

4.3.1 Modification Algorithm

Take the network in Section 4.2.2 as an example. Suppose a fault occurs between bus 4 and bus 6. Denote G_{pre}, G_{on} as conductance matrices in the pre-fault stage and fault-on stage, respectively. The construction of G_{pre}, G_{on}, G_{pre}^{-1} and G_{on}^{-1} is illustrated in Fig. 4.4.

In the pre-fault stage, G_{pre} is constructed according to the network before the fault occurs, shown in Fig. 4.4 (a). We hierarchically approximate G_{pre}^{-1} by Algorithm 4. Name the approxima-

tion of G_{pre}^{-1} as $\widetilde{G_{pre}^{-1}}$, shown in Fig. 4.4 (b).

In the fault-on stage, G_{pre} is updated to G_{on} by modifying corresponding elements in G_{gg}, G_{ii}, and H_{gi} due to the fault occurred between bus 4 and bus 6. Rename the modified block matrices as $G_{gg}^{'}, G_{ii}^{'}, H_{gi}^{'}$, shown in the Fig. 4.4 (c). Since only three blocks are modified, while others stay the same. According to Algorithm 4, $\widetilde{G_{on}^{-1}}$ is formed by modifying $(G_{gg}^{'})^{-1}, (G_{ii}^{'})^{-1}$, and $M_{gi}^{'}$ only, shown in Fig. 4.4 (d).

As described, the modification of G^{-1} updates a relative small portion of sub-blocks and avoids re-construct the entire G^{-1}. The blocks which need to be updated are easily located by the partition tree. In this example, bus 4 belongs to node g, and bus 6 belongs to node i. Starting from node g and i, we trace up to their ancestors until reaching their lowest common ancestor (LCA) which is node j. Thus node j is the smallest group that contains both bus 4 and 6, and the smallest group that is directly influenced by the fault. Therefore, the interaction within node j needs to be modified, thus all matrices that describe interactions between nodes along the paths from g to j and from i to j need to be modified. Since node g and i are leaves, the interaction within node g and i which represented as $(G_{gg}^{'})^{-1}$ and $(G_{ii}^{'})^{-1}$ are computed directly by inversion. The interaction between node g and i represented as $M_{gi}^{'}$ is computed by (4.5). The interaction within node j is approximated by interaction within node g, node i and between node g and i by (4.2). The tree with modification nodes is shown in Fig. 4.5, where the grey nodes are updated by directly inverse, the orange node is the LCA of grey nodes estimated by combining grey nodes.

The detailed algorithm is presented below. Suppose the network has been partitioned and G^{-1} has been approximated by Algorithm 4 in the pre-fault stage. The inputs are node i and j, where node i contains the fault bus i, node j contains the fault bus j.

The computation cost of the fast G^{-1} modification approach highly depends on the fault location. In the best case, the LCA of node i and j is itself, so that we don't need to update any M matrices, just update the diagonal matrix locally. In the worst case, the LCA of node i and j is the root, so that besides updating the diagonal matrix locally, we also need to update multiple M matrices from leaf to root.

Algorithm 5: Modification(i, j)

```
  // Find the LCA of node i and j, name it as node k
1 k = FindLCA(root, i, j);
  // Updating M recursively
2 UpdateM(k, i, j);
3 return
4 Function FindLCA(root, i, j):
5     if i, j ∈ {root.left.data} then
6         return FindLCA(root.left, i, j);
7     else if i, j ∈ {root.right.data} then
8         return FindLCA(root.right, i, j);
9     else
10        return root;
11    end
12 Function UpdateM(k, i, j):
13    if k is a leaf in the partition tree then
14        if k is the same as i or j then
15            direct compute the G_kk^{-1};
16        end
17        return
18    end
19    UpdateM(k.left, i, j);
20    UpdateM(k.right, i, j);
21    Update M_lr^{-1} by (4.7), where l and r are the left and right child of k, respectively;
22    return;
```

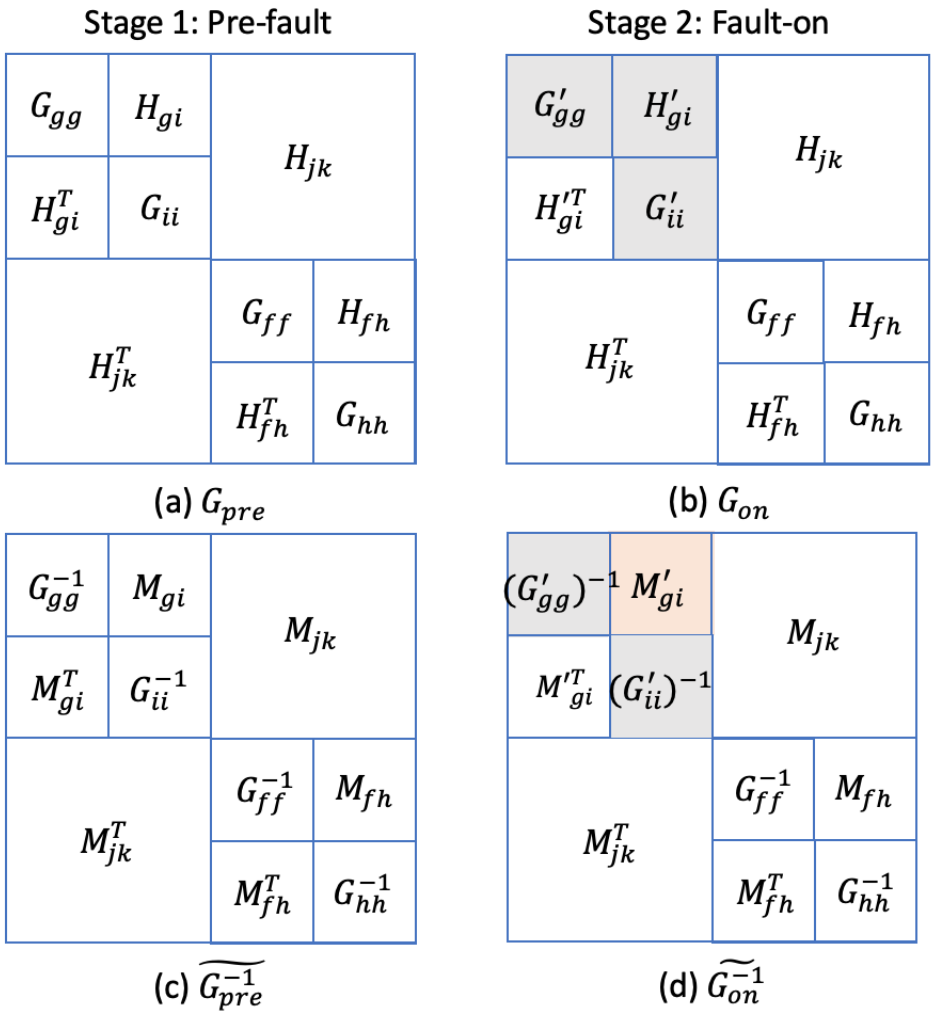


Figure 4.4: Modification of G^{-1} under a fault between g and i.

4.3.2 Time Complexity

Theorem 4. *Given an N-bus power system which is hierarchically partitioned, the time complexity of fast G^{-1} modification algorithm based on hierarchical approximation for G inversion is $O(n^2)$, where n is the smallest group of buses which contain buses directly connected to the fault.*

Proof. Suppose the N-bus power system network is partitioned into multiple groups, thereby produces a binary tree where the root represents the whole network, the leaves represents small groups of buses where each group contains no more than d_{th} buses. Therefore, the partition tree has $O(N)$ nodes and the height of the tree is $O(\log N)$.

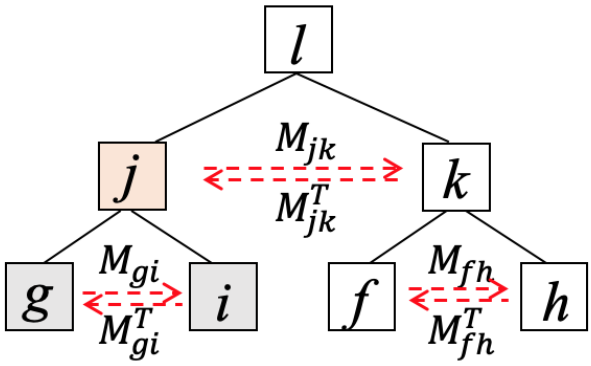


Figure 4.5: The updated tree of G^{-1} under a fault.

Consider when a fault occurs between bus i and j, which belongs to leaves i and j in the partition tree, respectively. Let node k be the lowest common ancestor of leaves i and j and let the number of buses under node k be n. According to Algorithm 5, the time complexity of fast G^{-1} modification algorithm only depends on n. Let $T(n)$ be the time complexity of fast G^{-1} modification algorithm under a fault, then we have

$$T(n) \leq \begin{cases} C & \text{if } n \leq d_{th} \\ 2T(\frac{n}{2}) + rn^2 & \text{otherwise,} \end{cases}$$

where C is a constant, $T\left(\frac{n}{2}\right)$ is the time complexity of updating diagonal matrices due to the partition, and rn^2 is the upper bound on time complexity of updating the off-diagonal matrices M, r is the number of edges between two groups.

To solve the recurrence relation, we have

$$\begin{aligned} T(n) &\leq 2T\left(\frac{n}{2}\right) + rn^2 \\ &= 2\left(2T\left(\frac{n}{4}\right) + r\left(\frac{n}{2}\right)^2\right) + rn^2 \\ &= \cdots \\ &= \frac{n}{d_{th}}T(d_{th}) + rn^2 \sum_{i=0}^{\log(n/d_{th})} \frac{1}{2^i} \\ &< \frac{n}{d_{th}} + 2rn^2 \\ &= O(n^2). \end{aligned}$$

The sum to $\log n/d_{th}$ is due to the fact that the length of the path from node i to node k, and from node j to node k is at most $\log n/d_{th}$.

□

4.4 Case Study

This section presents comprehensive case studies for EMT simulation on 179-bus based systems to evaluate the performance of the proposed approaches. The data and one-line diagrams of the 179-bus system can be found in [31]. In the simulation, Algorithm 4 is used to approximate G^{-1}. If a fault occurs, Algorithm 5 is used to update G^{-1}. For the network solution, the low-rank approximation based approach is adopted to further compress G^{-1}, then fast matrix-vector multiplication [41] is applied to solve the network equations.

4.4.1 Node Threshold v.s. G Inverse Accuracy

In this test, G matrix for a 179-bus system is used for evaluating the impact of the node threshold d_{th} on the accuracy of G^{-1}. In Fig. 4.6, as d_{th} increases from 2 to 74, the error exponentially drops from 2.5×10^{-3} to 1.8×10^{-9}.

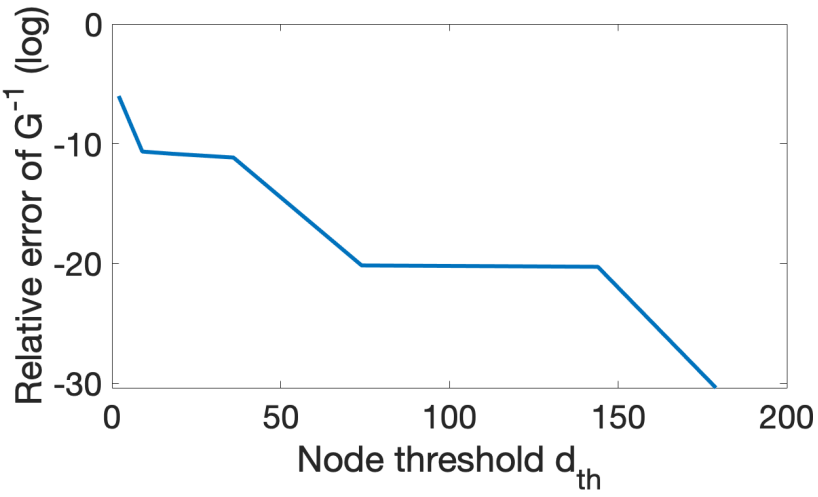


Figure 4.6: Relative error of G^{-1} with node threshold d_{th}.

4.4.2 CPU Time in terms of FLOPS

In this test, a series of large systems consist of multiple copies of a 179-bus system interconnected as an array. A similar system is used in [12]. The FLOPS involved in approximate G^{-1} by Algorithm 4 and the network solution by fast matrix-vector multiplication [41] at one time step are counted. The FLOPS count includes total numbers of floating point addition and multiplication.

4.4.2.1 FLOPS for Hierarchical Approximation of G inverse

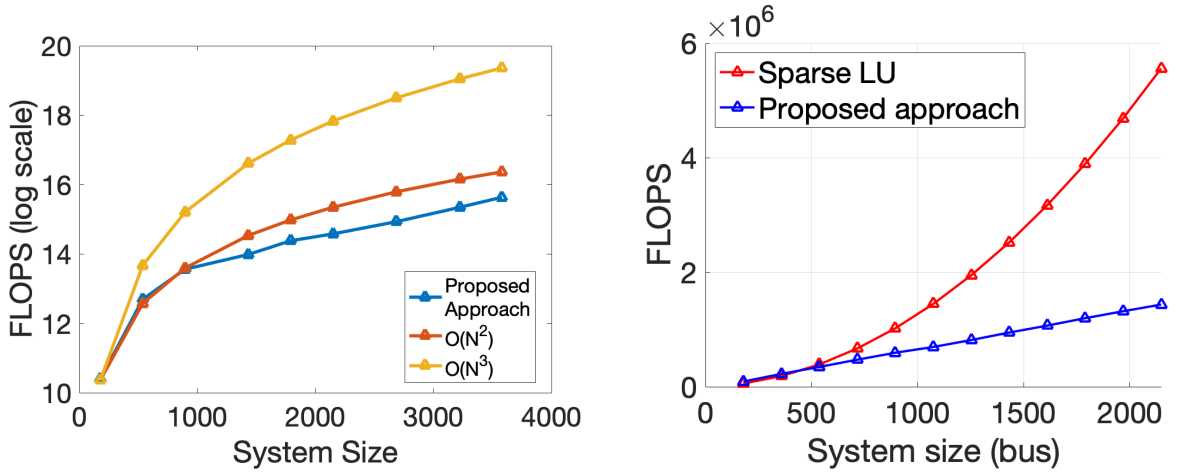
Traditionally, the time complexity of matrix inverse is O(N^3), where N is the system size. Fig. 4.7(a) shows that in the series of 179-bus based systems, the FLOPS of Algorithm 4 is O(N^2).

4.4.2.2 FLOPS for Network Solution

Fig. 4.7(b) shows that in the series of 179-bus based systems with up to 2148 buses, the maximum FLOPS of LU-based approach is 5.56×10^6, while the fast fatrix-vector multiplication [41] is 1.43×10^6, leading to an 74% reduction compared with LU-based approach.

4.4.3 EMT Fault Simulation

In this test, a three-phase ground fault on a 179-bus system is simulated with a 20μs time step and a total simulation time length of 60ms. The fault between bus 1 and bus 81 is added at 10ms



(a) Flops for G^{-1} computing

(b) Flops for network solution

Figure 4.7: FLOPS comparison.

and lasts for 20ms. The threes grounding resistances are 10 Ωs. The partial network with the fault is shown in Fig. 4.8. To obtain a better accuracy, the node threshold d_{th} is set to 74 at beforehand.

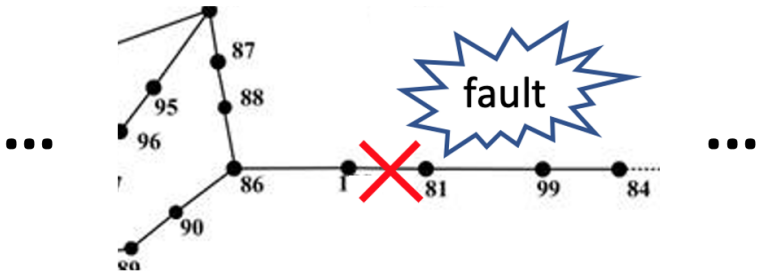
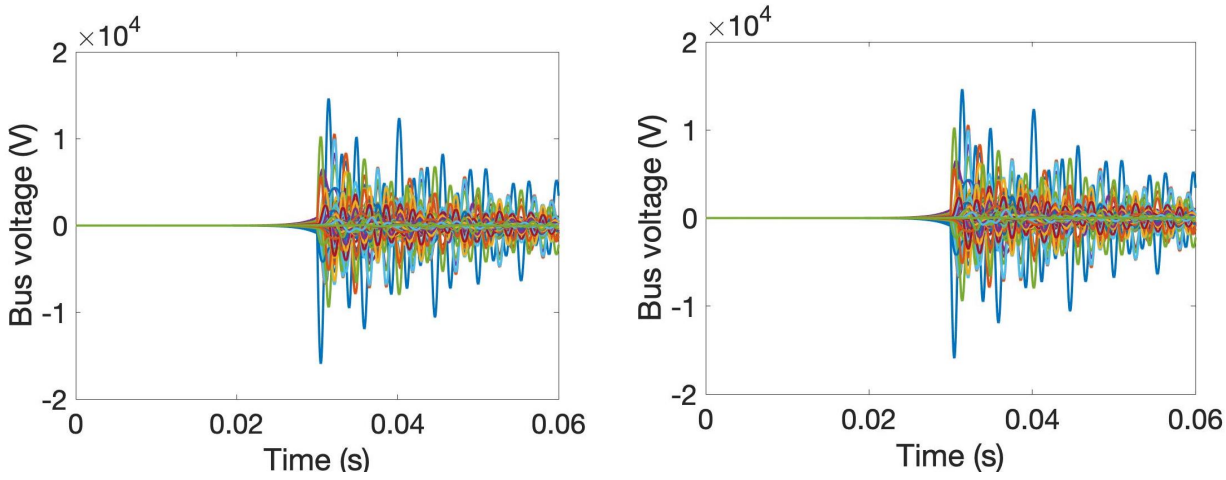


Figure 4.8: The 179-bus system with a fault.

4.4.3.1 Accuracy Analysis

An LU-based network solver is used as a reference. The relative error of the bus voltage is measured at each simulation step. As shown in Fig. 4.9 and 4.10, the result by the proposed approach matches well with the reference. The maximal relative error of bus voltage is 7.4×10^{-5}.



(a) LU-based Approach (b) Proposed approach

Figure 4.9: Accuracy comparison.

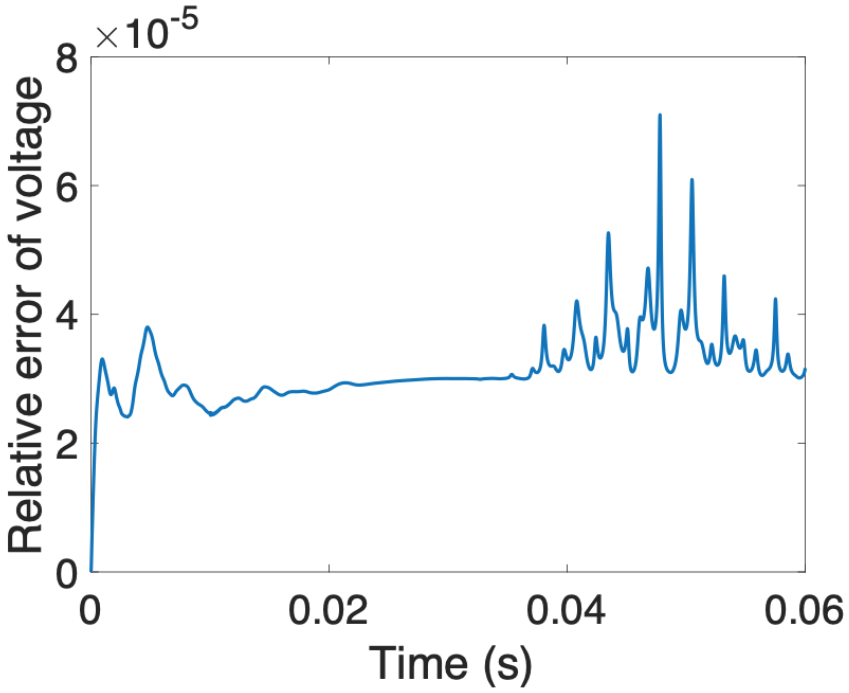


Figure 4.10: Relative error of bus voltage on 179-bus system.

4.4.3.2 Computation Cost for G Inverse Modification

The modification cost of fast G^{-1} modification approach depends on the fault location. In this case, the fault occurs in the same leaf node of the partition tree. Since the node threshold is set to 74, the proposed approach only needs to invert the 74×74 matrix. Compared with inverting the entire 179×179 matrix, this approach reduces 82.9% computation.

4.5 Conclusion

This chapter presents an inverse-based scheme to solve the network. To break down the bottleneck of computing sparse matrix inversion, this chapter proposes a hierarchical approximation of G^{-1} by first hierarchically partitions the network into multiple groups, then make use of the local impacts within groups to approximate the interaction between groups which significantly reduces the computational time. Further, a fast G^{-1} modification approach is presented for contingency analysis. The performance is demonstrated by the EMT simulation on 179-bus based systems. The experiments show that the proposed approaches exhibit great advantage in speeding up the network solution with high accuracy.

5. VLSI CIRCUIT DESIGN

The highly parallel scheme algorithm exhibits a great potential for speeding up the EMT simulation no matter what hardware implementations are applied. It could be feasible for multi-CPU, multi-GPU/FPGAs (Field Programmable Gate Array), cloud computing, ASIC (Application-Specific Integrated Circuits), etc. In this chapter, the most challenging and high-performance implementation which is a full-custom ASIC design will be exploited, since it will fully utilize the parallel algorithm and fundamentally speed up the computation by leveraging the unique architectural capability for the EMT simulation.

5.1 Background

Very-large-scale integration (VLSI) is the process of creating an integrated circuit (IC) by combining thousands of transistors into a single chip. It began in the 1970s when complex semiconductor and communication technologies were being developed. With the advent of VLSI designs, the number of applications of ICs in high-performance computing, controls, telecommunications, image and video processing, and consumer electronics has been rising at a very fast pace. Among the VLSI, there are two typical types of circuit: FPGA and ASIC. FPGA is an integrated reconfigurable and programmable circuit which can be "field" programmed to work as different the intended design. They even have capability to reconfigure a part of chip while remaining areas of chip are still working. This feature makes FPGAs highly used for applications such as radars, cell phone base stations etc., where the current design requires upgrade frequently. Currently, it is also the most adopted approach used in the hardware-based EMT simulation. However, this implementation has an inherent limitation that the off-the-shelf FPGA architecture is designed for general purpose random logic circuit applications, thus inefficient when used for numerical simulation due to the wasting of hardware resources, increasing signal propagation delay between multiple logic cells. In contrast, ASIC is an integrated circuit developed for a particular use. It is a permanent circuitry which means once the application specific circuit is taped-out into silicon, it cannot be changed

or re-programmed. However, just due to the specific design, the ASIC architecture exhibits great numerical computation performance with much less hardware resources with an optimized design making more efficient than FPGAs.

5.2 ASIC Architecture

Architecture design is an essential component which is critical to performance and cost in the ASIC design. It describes the whole system structure of the simulation, the identification of the system's physical components and their interrelationships.

The proposed high performance full-custom ASIC architecture will follow five principles: modularity, hierarchy, encapsulation, regularity, and extensibility. Matched to the domain of interest, we map our algorithm into multiple modules and reclusively use modules to develop a hierarchical structure based on the relationship. Each module will define an interface that hidden the detail design inside the module and expose interconnections with other modules. For the sake of further changes, we will include some mechanisms or hooks to enable more robust composition for the whole architecture. In particular for the EMT simulation, we decompose the whole EMT simulation into several high-level abstract modules including generators, exciters, control elements and network with well-defined interfaces. To accelerate the computation, a massively parallelizable scheme is proposed in Fig. 5.1, where all state variables of generators, control elements, and loads are updated concurrently in parallel and state variables of multiple well-separated groups of buses by low-rank approximation are updated concurrently in parallel in network so that the speed of large-scale system EMT simulation is efficiently increased.

5.3 Data Flow for Hardware Implement

According to the draft simulation flow in Fig. 2.9 and algorithms we proposed, the simulation flow is reorganized in a hardware friendly scheme, depicted in Fig. 5.2 where each box is mapped into a module in ASIC design.

Fig. 5.2 shows that each generator solver contains seven high-level modules, each control element solver contains one high-level modules, and each network solver contains two high-level

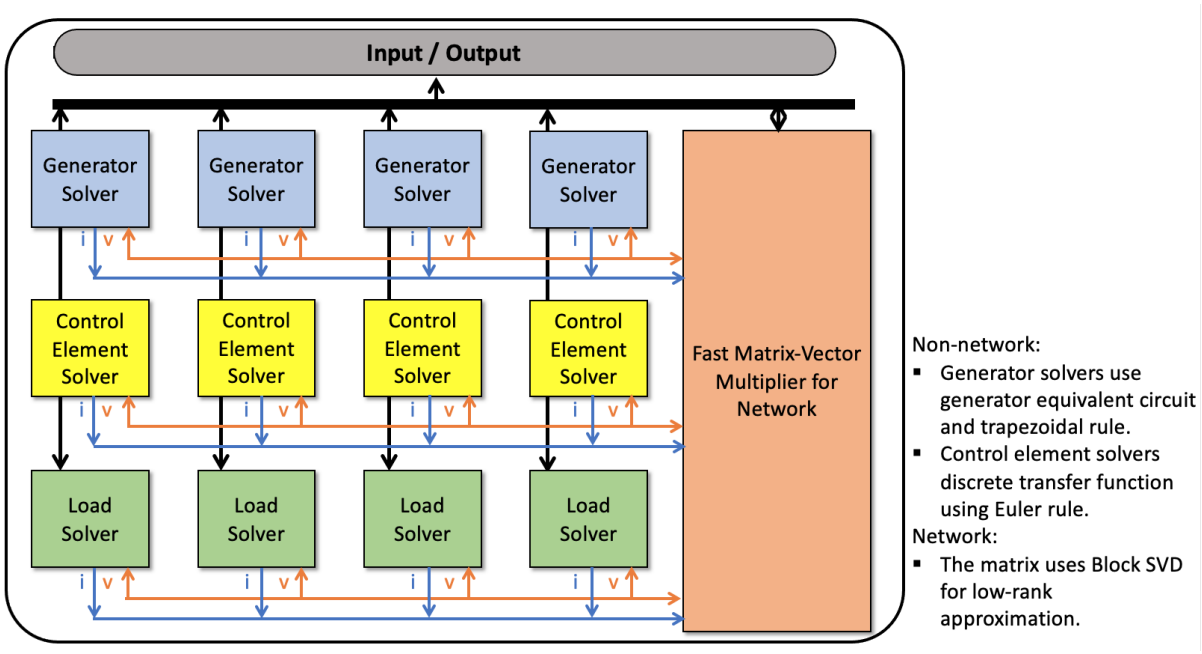


Figure 5.1: Proposed architecture for the ASIC.

modules. Note that, since the constant impedance load model is adopted in the simulation, we simplified the flow by merge the load into network topology. All the generators are solved in parallel, then pass the current to network module to solve node voltages through the interface park transformation and inverse park transformation. The data flow between each modules is depicted in Fig. 5.3.

Besides the coupling relationship between different modules, there are a few things to be taken into account as well.

1. **Initialization**. To passing parameters and initial values of the registers from the PC side to ASIC, multiple scan chains are taken into consideration.

2. **Park and inverse park transformation**. The park transformation is a sinusoidal matrix vector multiplication where the sinusoidal matrix depends on angel θ between abc coordinate and DQ0 coordinate. Thereby, the sinusoidal value needs to be updated in each time step. In order to speed up the computation and maintain high accuracy, we design a $64K \times 32$ ROM to directly map an input address in the range of 0 to $pi/2$ to its sinusoidal value.

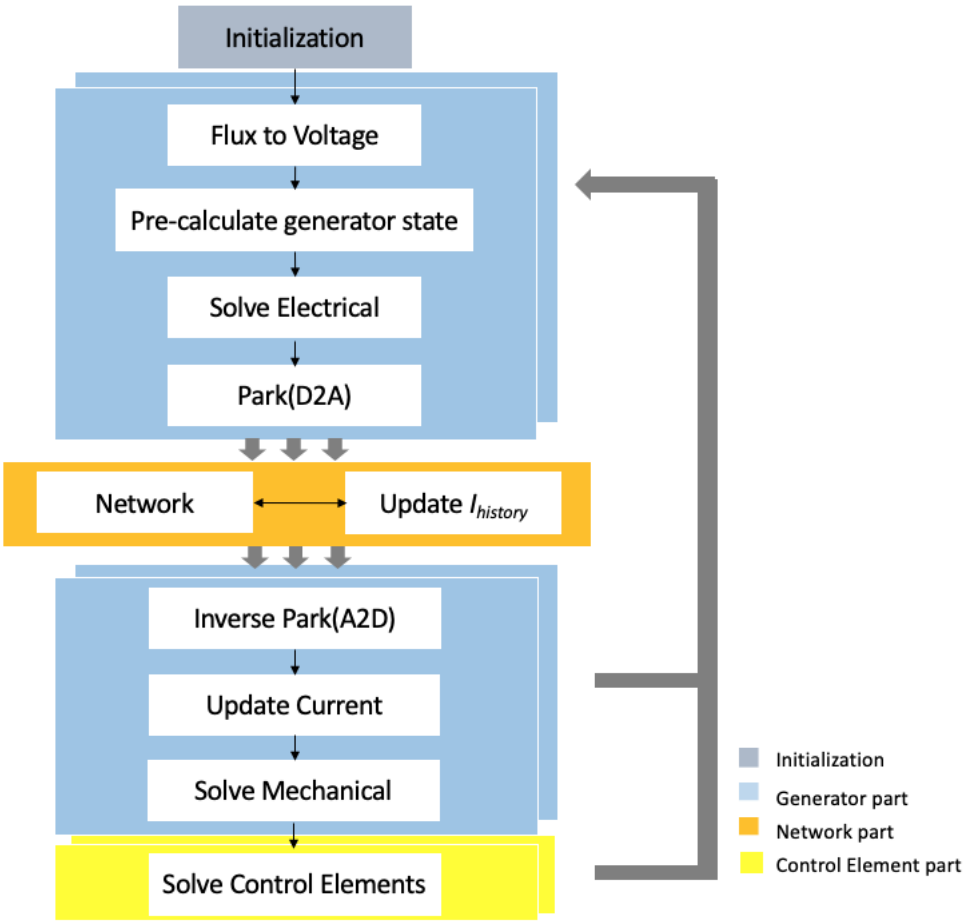


Figure 5.2: The EMT simulation flow using nodal analysis.

5.4 Hierarchical Design

Following the five principles in Section 5.2, we hierarchically decompose the high-level modules such as generators into multiple submodules based on the different functions to be implemented until the finest modules are standard cells such as adders, multipliers, shifters in the library. In the end, the system architecture the EMT simulation is divided into five levels as shown in Fig. 5.4. The bottom level consists of modules such as adders, multipliers, shifters and matrix-vector multipliers. The next higher level consists of ODE solver, linear equation solver, etc. The next higher level consists of function modules such as pre-calculation, park transformation, etc. Above the function module level is the component level, consisting of generator machine models and the

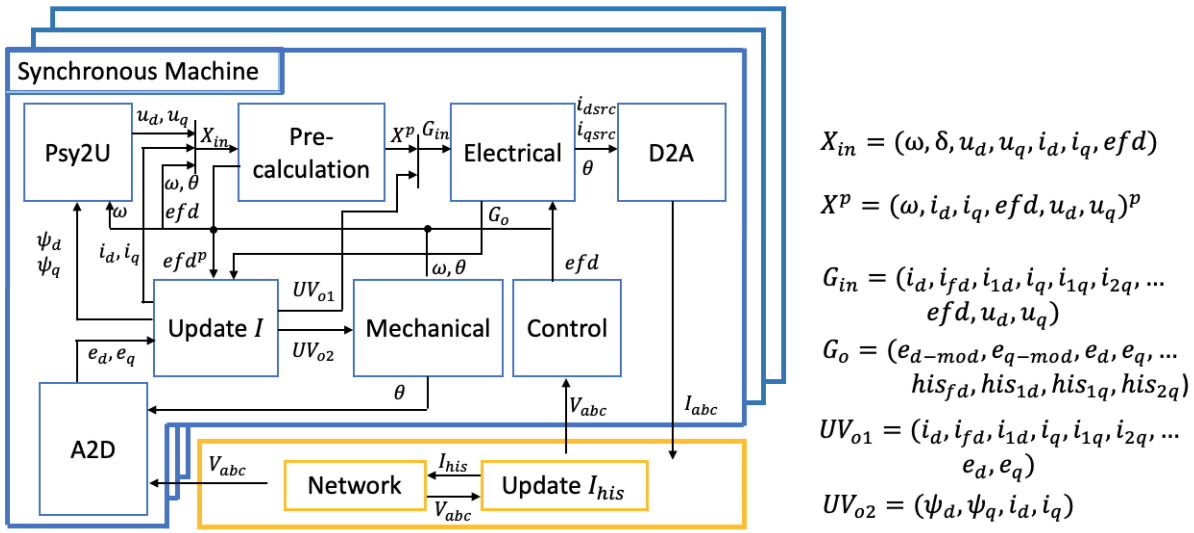


Figure 5.3: Data flow of the EMT simulation.

transmission line model. The top level connects all components together. Deep analyze the algorithm and optimized decompose structure into varies abstract level, thus maximum the capability of reusable and reduce the complexity of ASIC design.

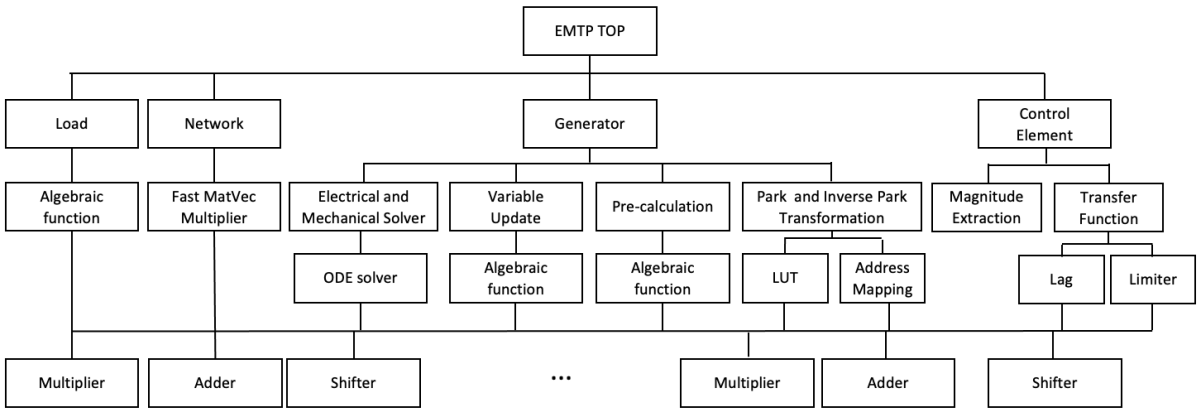


Figure 5.4: Hierarchical architecture of the EMT simulation.

Our proposed algorithm will be map into architecture by the following steps.

1. Allocate the multiple functional units in algorithm into the corresponding behavior modules

with well-defined interfaces. For example, the low-rank approximation algorithm will be mapped into the network module and the interface will be the node current vector I and node voltage vector V.

2. Design logic control blocks, organize n-bit datapath and custom design the algebraic equations by basic arithmetic operations like multiplier, adders. For example, use 2-bit right shifter to implement the coefficient 0.25.

3. Fully exploit the parallelism and utilize the pipeline design to speed up runtime.

5.5 EDA Software and Process Technology

To create high-level representations of a circuit, the system is designed by Verilog HDL using a TSMC 180nm process technology. The EDA tools that are used in the design are listed in the below.

1. NC-Verilog and SimVision are used as verilog compiler and simulator.

2. Synopsys Design Vision/Compiler which is the graphical interface is used as logic synthesis.

3. Synopsys Primetime is used for pre-layout static timing analysis.

4. Cadence SoC Encounter which a place-and-route tool that uses a verilog netlist and generates its equivalent layout view is used for automatic place-and-route.

The data width of all variables is 64 bits fixed-point, with 1 bit sign, 31 bits for integer and 32 bits for fraction.

5.6 Initialization

Before the start of the simulation, three types of initial values are required to load into registers.

1. Model parameters including resistances, capacitance, time constants, or other coefficients of the algebraic equations, etc.

2. System initial state values including power flow data, simulation setting, etc.

3. Rom memory values which stay constant during the entire simulation.

Once the system parameters and initial values are ready, these external data are shifted into the registers through a scan chain at the initialization. To reduce the time to load and observe, multiple scan chains are built in parallel. In general, the number of scan chains depends on the number of ports required on the chip, e.g., number of scan chains equals half number of ports required.

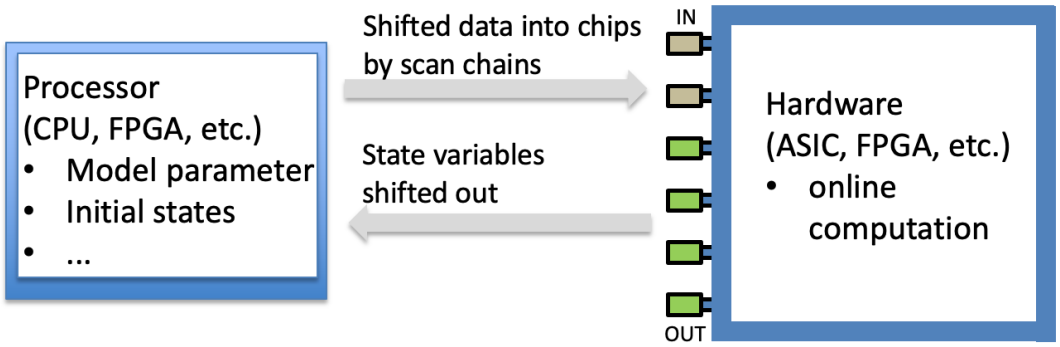


Figure 5.5: System structure of the EMT simulation.

5.7 Data Prefetching in EMT Contingency

The power system is a interaction network where all buses connected and interacted with each other directly or indirectly. Therefore, once a fault occurs on a transmission line, all buses will be affected. Wait the updating of entire G^{-1}, then do matrix-vector multiplication will take a lot of time. Therefore, this section introduce a data prefetching scheme which is a technique to boost execution performance by fetching data from their original storage in slower memory to a faster local memory before it is actually needed for G^{-1} to speed up the EMT contingency analysis.

5.7.1 Partial Updating of G Inverse

The EMT fault simulation can be divided into three stages, e.g., pre-fault stage, fault-on stage, post-fault stage, according to the time of failure, each stage has a corresponding G and G^{-1}. To

clearly illustrate how G and G^{-1} changes during the fault, let's take an example. Suppose a three-phase-to-ground fault occurs between bus m and n, and clears at some time later. We denote G_{pre}, G_{on}, G_{post} are the G matrices in the pre-fault, fault-on, and post-fault stage, respectively. Since the entries G_{mn} of impedance matrix represents the relationship between bus m current and bus n voltage, we need to modify entries $G_{ii}, G_{ij}, G_{ji}, G_{jj}$ over three phases, i.e. total is 36 elements, when fault occurs or clears. The changing of G during the fault is shown in Fig. 5.6, where the red dots in matrices are the entries need to be modified.

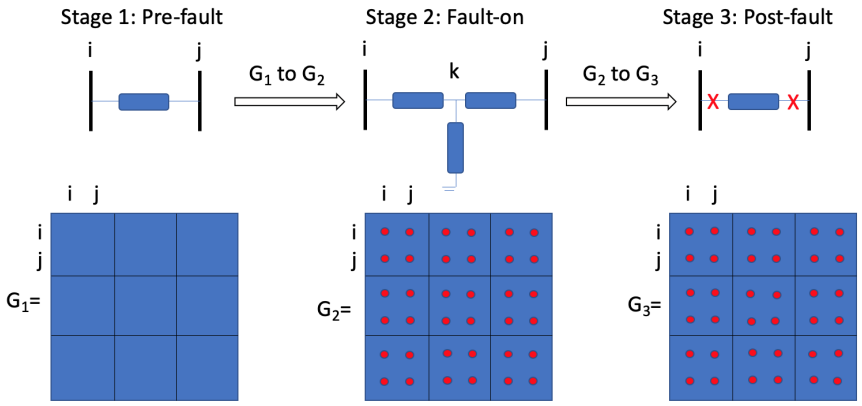
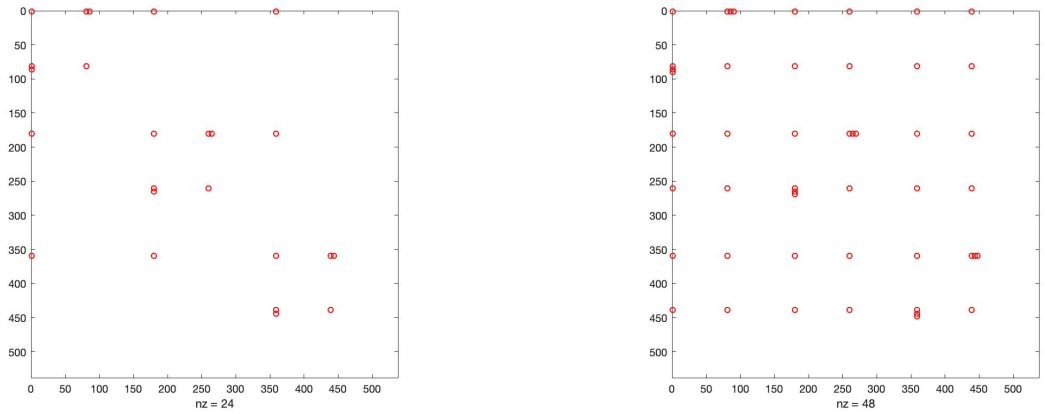


Figure 5.6: The changing of G matrix during the fault.

Due to the modification of G, the G^{-1} is changed correspondingly. Unlike the modification of 36 entries in G, all the entries in G^{-1} have to be updated, which cost lots of computations and memory. Thus, to save computation by minimized number of modification, we set a threshold θ to filter out entries G^{-1}_{ij} that if the changing amount is greater than θ, we modify it, otherwise, we keep it. Comprehensive testing results shows that most of entries G^{-1}_{ij} are changed within a relative small amount. Therefore, by setting a reasonable θ, we could ignore most of changing entries and only update limited entries.

To verify the observation and evaluate the influence of different θ, we adopt the G^{-1} approximation to do the EMT simulation on the 179-bus system, and check the accuracy. The detail simulation settings are same as setting in Section 4.4.3.

Fig. 5.7 depicts the elements need to be modified between G_{pre}^{-1} and G_{on}^{-1} where the changing amount is greater than tolerance θ.



(a) θ is 1e-7 (b) θ is 1e-8

Figure 5.7: Differences elements between G_{pre}^{-1} and G_{on}^{-1} greater than tolerance.

Fig. 5.7 shows that

1. when $\theta = 10^{-7}$, 24 elements need to be modified from G_{pre} to G_{on}. All the differences locate at (i, j), where $i, j \in \{1, 81, 86\}$, where bus 86 is the neighbor of bus 1, saved 99.99% elements compared with original 288369 elements.

2. when $\theta = 10^{-8}$, 48 elements need to be modified from G_{pre} to G_{on}. All the differences locate at (i, j), where $i, j \in \{1, 81, 86, 90\}$, where bus 86 is the neighbor of bus 1, and bus 90 is the neighbor of bus 86, saved 99.98% elements compared with original 288369 elements.

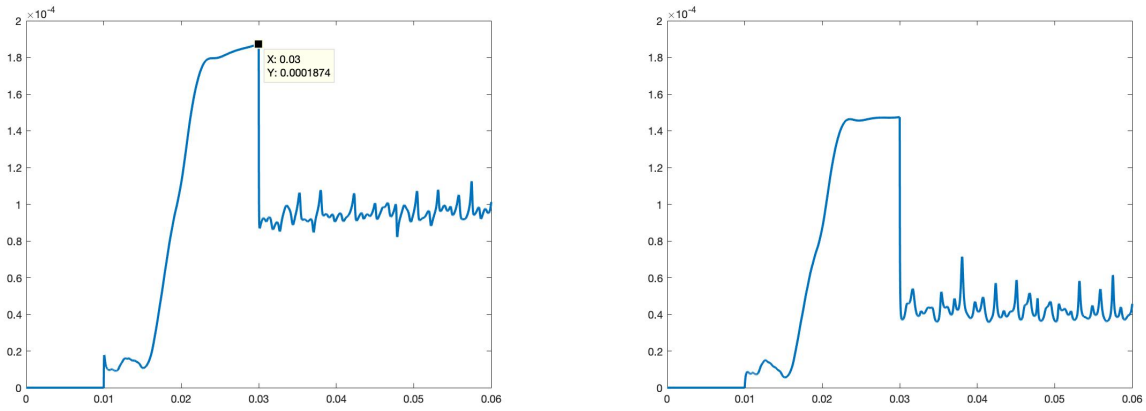
The example provides an evident that

1. given a small tolerance threshold, only a small portion of elements in G^{-1} need to modified when fault happens.

2. the majority modified impedance locate on the transmission lines connected to the fault buses or its neighbors.

The observation also implies the fact that the fault has a strongest and direct impact on the buses connected to the fault line, then the impact propagates through transmission lines to other buses.

Fig. 5.8 shows the relative error of bus voltage when apply the approximation G^{-1} with different θ. As shown, the largest relative error of bus voltage is as small as 1.8×10^{-4} and 1.5×10^{-4} when $\theta = 10^{-7}$ and $\theta = 10^{-8}$, respectively.



(a) θ is 1e-7

(b) θ is 1e-8

Figure 5.8: Relative error of bus voltage using approximation G^{-1} with θ.

5.7.2 Incremental Partial Updating of G Inverse

The fault impact follows a pattern that the strongest influences happen at buses directly connect to the fault transmission line, then the influences decade through the transmission lines to the whole network. The partial updating test also demonstrates the observation. Therefore, according to the straighten of impact, we update state of buses in multiple stages when fault happens.

Suppose a fault occurs between bus i and j. The updating follows the order shown in Fig. 5.9.

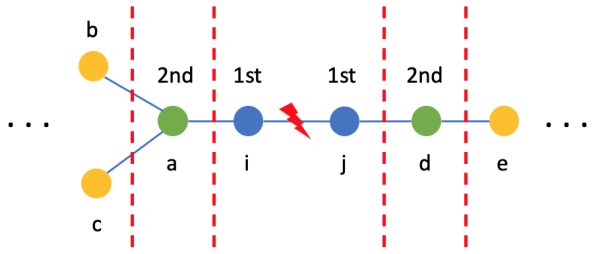


Figure 5.9: Multiple steps incremental updating scheme.

1. Update bus i and j since these two has strongest impact by a fault;

2. Update the neighbors of bus i and j which are bus a and d;

3. Update the neighbors of bus a and d which are bus b, c and e;

4. Keep update neighbors of the previous updated buses until all buses are visited.

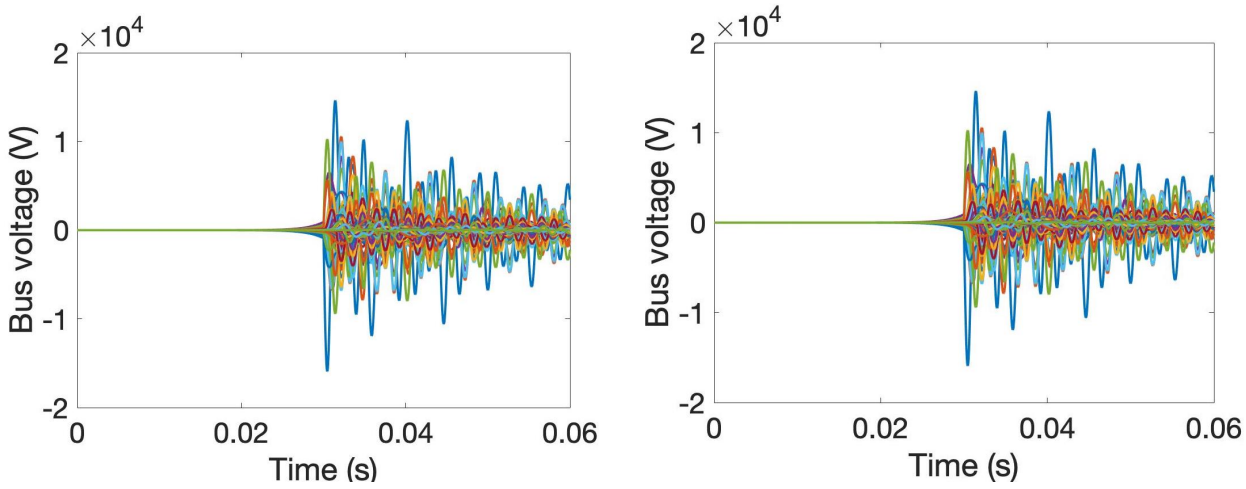
Section 5.7.1 demonstrates that buses around the fault have strongest impact on the network, thus two- or three-stage incremental updating is adopted to modify G^{-1} in the EMT contingency which depends on the requirement of accuracy.

Take the same system and fault simulation setting in Section 4.4.3 as an example. Take accurate entire updating G^{-1} as a reference. We update G^{-1} with two-stage incremental updating. In the fault-on stage, states of bus 1 and 81 are updated, thereby elements (1, 1), (1, 81), (81, 1), (81, 81) in G^{-1} are updated. Then states of bus 86 and 99 are updated, thereby elements (86, 86), (86, 99), (99, 86), (99, 99) in G^{-1} are updated. The simulation result is shown in Fig. 5.10, and the relative error of bus voltages is shown in Fig. 5.11.

As Fig. 5.10 shown, the result matches the reference well. Fig. 5.11 shows the relative error of $|G|$, where the largest error turns out to be as small as 6.5×10^{-5} A.

Test case with approximation G^{-1} by Algorithm 5 is presented in Fig. 5.12, and the relative error of bus voltages is shown in Fig. 5.13.

As Fig. 5.12 shown, the result also matches the reference well. Fig. 5.13 shows the relative error of $|G|$, where the largest error turns out to be as small as 6.5×10^{-5} A.



(a) Reference approach (b) 2-step incremental partial updating approach

Figure 5.10: Accuracy comparison with accurate G^{-1} updating.

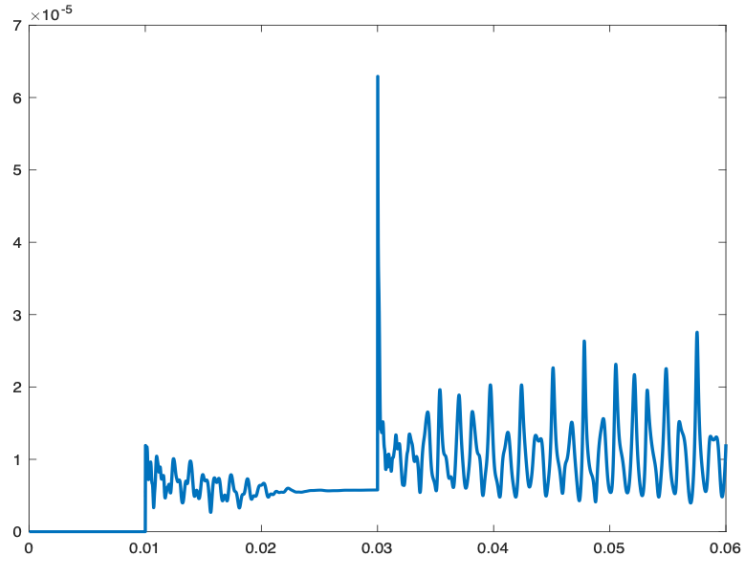
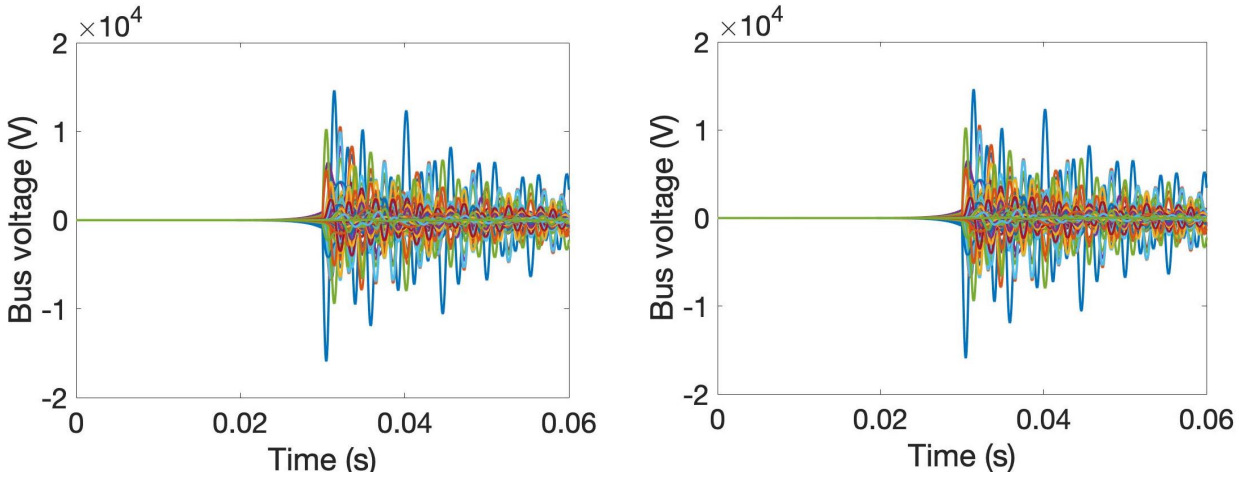


Figure 5.11: Relative error of bus voltage with 2-step incremental partial accurate G^{-1} updating.

These test cases demonstrate that the incremental partial updating scheme is applicable for EMT simulation without scarified much accuracy. Therefore, once a fault happens, the system doesn't have to wait for the updating of entire G^{-1}, while the system can update the G^{-1} in multiple



(a) Reference approach (b) 2-step incremental partial updating approach

Figure 5.12: Accuracy comparison with approximation G^{-1} updating.

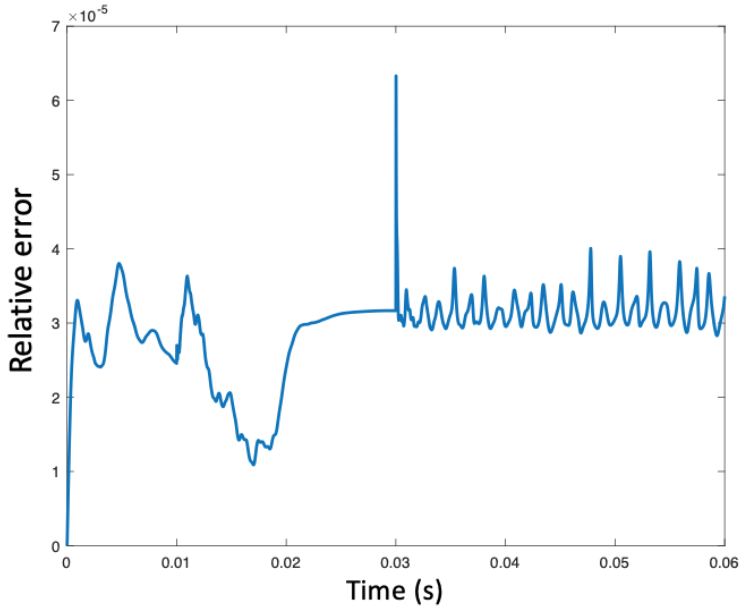


Figure 5.13: Relative error of bus voltage with 2-step incremental partial approximation G^{-1} updating.

time steps where in each time step only partial elements are updated. The faster memory can store the date for next time step in beforehand to save the computation time.

5.8 RTL Design

Fig. 5.4 shows that the design is built by multiple levels where each level consists of multiple modules. In this section, we focus on the register-transfer level (RTL) design of the sub-level of the components, e.g., generator, exciter, network. The RTL is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers, and the logical operations performed on those signals. In our design, we use verilog HDL to create high-level representations of a circuit, then use advanced EDA tool to derive well-defined low-level representations automatically.

5.8.1 Generator Solver

5.8.1.1 Convert Flux Linkage to Speed Voltage

The module which converts Flux linkage to speed voltage is named as "Psy2U" in the design. The equation to be implemented is shown in (5.1), and the corresponding designed circuit is shown in 5.14.

$$\begin{cases} u_d = -\psi_q \frac{\omega}{\omega_b}, \\ u_q = -\psi_d \frac{\omega}{\omega_b}, \end{cases} \tag{5.1}$$

where ψ_d and ψ_q are flux linkage on the direct axis (d-axis) and quadrature axis (q-axis), respectively; u_d and u_q are speed voltage on the direct axis and quadrature axis, respectively; ω is the rotor angle speed, and ω_b is the base of rotor angle speed which is a constant.

Fig. 5.14 shows that the designed circuit is a combination circuit whose output at any instant in time depends only on the combination of its inputs. Two adders and one multiplier are used in the design.

5.8.1.2 Pre-calculate the Generator States

To avoid numerical instability issue of the EMT, the nodal analysis adds one step to pre-calculate some generator variables x and rotor angle δ based on the current system states so that we can assume the generator part and network part are simulated simultaneously.

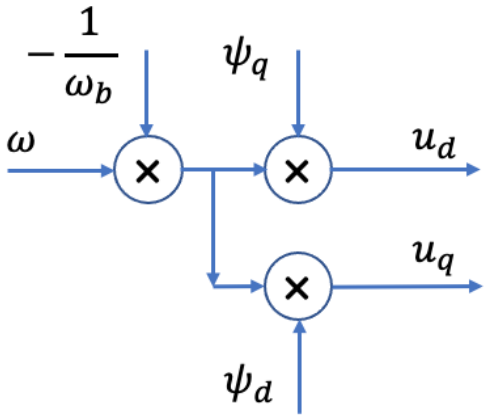


Figure 5.14: Circuit for "Psy2U" module.

Since the generator variables x are not changing abruptly, their values can be reasonable pre-calculated through the rolling windows-based regression in the next time $t+1$ using not only $x(t)$, but also $x(t-1)$ and $x(t-2)$, shown in (5.2). The equation shows that when $t \leq 1$, no enough historical terms can be adopted. Therefore, we compute the first two steps in the offline, then shift-in the states in the third time step as initial values.

$$x^p = \begin{cases} x(t) & t = 0, \\ 2x(t) - x(t-1) & t = 1, \\ 1.25x(t) + 0.5x(t-1) - 0.75x(t-2) & t > 1, \end{cases} \tag{5.2}$$

where $x = [\omega, i_d, i_q, e_{fd}, u_d, u_q]$, x^p is the pre-calculated x, i_d and i_q are terminal current on the d-axis and q-axis, respectively.

The rotor angle δ is pre-calculated through (5.3).

$$\delta^p = \delta(t) + 0.5\Delta t(\omega(t) + \omega^p). \tag{5.3}$$

The circuit of pre-calculation when $t > 1$ is shown in Fig. 5.15. The designed circuit is a sequential circuit whose output not only relies on the current input but also depends on the previous

output. Registers are used to store history terms. Three shifters are used to creating coefficients for each history terms instead of multipliers. Rest are five adders and one multiplier. The module takes one clock cycle to update output.

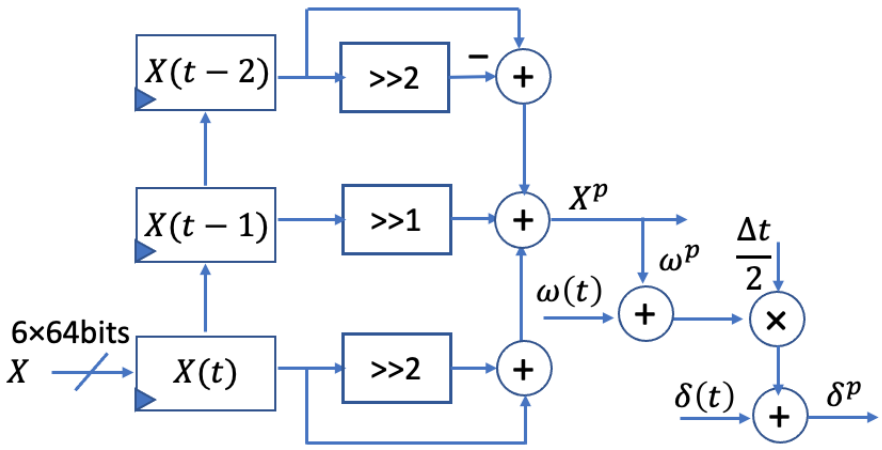


Figure 5.15: Circuit for "Pre-calculate" module.

5.8.1.3 Electrical Solver

Nodal analysis applies the trapezoidal rule of integration to the generator first, then reorganized the model by Norton equivalent circuit, and merge partial generator model into network. This reduced generator with the electrical equivalent circuit which named as "Electrical" is solved by (5.4). The corresponding circuit is shown in Fig. 5.16.

$$
\begin{aligned}
\begin{bmatrix} his_d \\ his_{fd} \\ his_{1d} \end{bmatrix} &= C_1 \begin{bmatrix} i_d \\ i_{fd} \\ i_{1d} \end{bmatrix} + c_1 \begin{bmatrix} u_d - e_d \\ e_{fd} \\ 0 \end{bmatrix}, \\
\begin{bmatrix} his_q \\ his_{1q} \\ his_{2q} \end{bmatrix} &= C_2 \begin{bmatrix} i_q \\ i_{1q} \\ i_{2q} \end{bmatrix} + c_2 \begin{bmatrix} u_q - e_q \\ 0 \\ 0 \end{bmatrix}, \\
e_{dmod} &= u_d^q + his_d - C_3 \begin{bmatrix} his_d & his_{fd} & his_{1d} & e_{fd}^p & i_d^p \end{bmatrix}^T, \\
e_{qmod} &= u_q^q + his_q - C_4 \begin{bmatrix} his_q & his_{1q} & his_{2q} & i_q^p \end{bmatrix}^T, \\
i_{dsrc} &= c_3 e_{dmod}, \\
i_{qsrc} &= c_4 e_{qmod},
\end{aligned}
\tag{5.4}
$$

where C_1, C_2 are 3×3 constant coefficient matrices; C_3, C_4 are 1×5 and 1×4 constant coefficient vectors, respectively; c_1, c_2,c_3, c_4 are constant scalar.

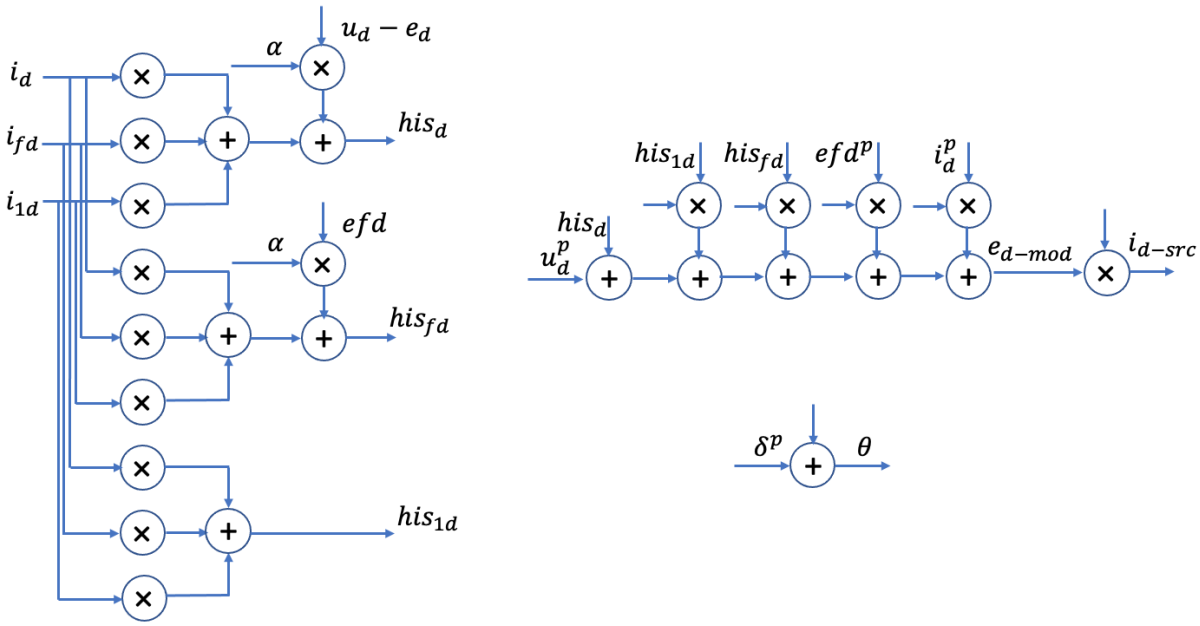


Figure 5.16: Circuit for "Electrical" module d-axis variables.

Consider about the timing requirement, a combination circuit is used in electrical solver which means the input reflects the output instantly. 6 three-way adders, 13 two-way adder, and 30 multiplier are used in the design.

5.8.1.4 Mechanical Solver

Besides electrical part which describes current and voltage on the d-axis and q-axis, the generator also consist of a mechanical part which describes the rotor angle and speed. The mechanical part in the generator is rewritten by (5.5) after discrete by Euler rule.

$$\begin{aligned} t_e(t+1) &= \psi_d(t+1)i_q(t+1) - \psi_q(t+1)i_d(t+1), \\ \omega(t+1) &= c_1(t_e(t+1) + t_e(t)) + c_2\omega(t) + c_3, \\ \delta(t+1) &= \delta(t) + c(\omega(t+1) + \omega(t)), \end{aligned} \tag{5.5}$$

where ω is the rotor speed, δ is the rotor angel, c_1, c_2, c_3, c are constant scalar.

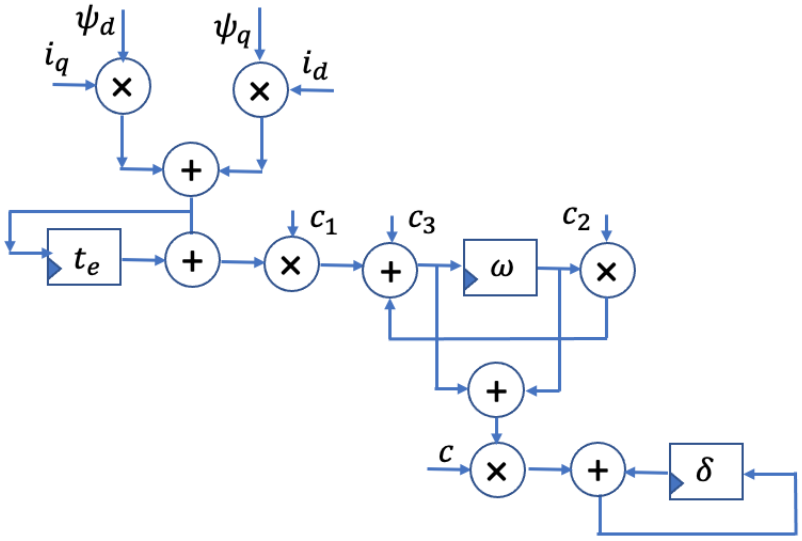


Figure 5.17: Circuit for "Mechanical" module.

Fig. 5.17 is the circuit for (5.5). The designed circuit is a sequential circuit which takes two clock cycle to update the output. Registers are used to store history terms. Rest are 5 adders and 5

multipliers.

5.8.1.5 Current Correction

This module is to correct generator terminal currents and winding currents in the d-axis and q-axis using history term and terminal voltages computed by network equation. The equation is shown in (5.6), and the corresponding circuit is shown in Fig. 5.18.

$$
\begin{aligned}
i_d &= c_1 e_{dmod} + e_d, \\
i_q &= c_1 e_{qmod} + e_q, \\
i_{fd} &= C_1 \begin{bmatrix} e^p_{fd} & his_{fd} & his_{1d} & i_d \end{bmatrix}^T, \\
i_{1d} &= C_2 \begin{bmatrix} e^p_{fd} & his_{fd} & his_{1d} & i_d \end{bmatrix}^T, \\
i_{1q} &= C_3 \begin{bmatrix} his_{1q} & his_{2q} & i_q \end{bmatrix}^T, \\
i_{2q} &= C_4 \begin{bmatrix} his_{1q} & his_{2q} & i_q \end{bmatrix}^T, \\
\psi_d &= C_5 \begin{bmatrix} i_d & i_{fd} & i_{1d} \end{bmatrix}^T, \\
\psi_q &= C_6 \begin{bmatrix} i_q & i_{1q} & i_{2q} \end{bmatrix}^T,
\end{aligned}
\tag{5.6}
$$

where C_1, C_2 are 1×4 constant vectors, C_3, C_4, C_5, C_6 are 1×3 constant vectors.

Consider about the timing requirement, a combination circuit is used in electrical solver which means the input reflects the output instantly. 18 two-way adder, and 24 multiplier are used in the design.

5.8.1.6 Park and Inverse Park Transformation

The park transformation and inverse park transformation are the interface between non-network and network. The park transformation applies a 3x3 sinusoidal matrix-vector multiplication to convert state variables between DQ0 coordinate in the generator side and three-phase coordinates in the network, while the inverse park transformation revert the conversion direction of park trans-

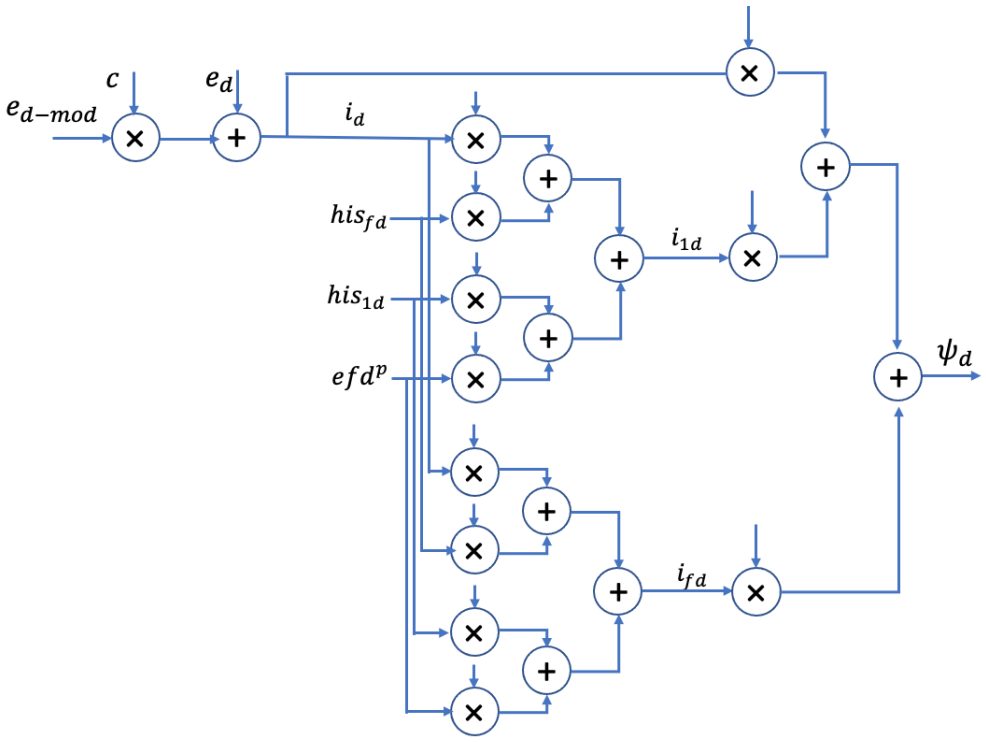


Figure 5.18: Circuit for "Current Correction" module d-axis variables.

formation. The equations for park transformation and inverse park transformation are described in (5.7) and (5.8), respectively.

$$\begin{bmatrix} I_a(t) \\ I_b(t) \\ I_c(t) \end{bmatrix} = P_i(\theta) \begin{bmatrix} i_d(t) \\ i_q(t) \\ 0 \end{bmatrix}, P_i(\theta) = \begin{bmatrix} sin(\theta + \frac{1}{2}\pi) & -sin(\theta) & 1 \\ sin(\theta - \frac{1}{6}\pi) & sin(\theta + \frac{1}{3}\pi) & 1 \\ sin(\theta + \frac{1}{6}\pi) & sin(\theta - \frac{1}{3}\pi) & 1 \end{bmatrix}, \tag{5.7}$$

$$\begin{bmatrix} e_d(t) \\ e_q(t) \\ \varepsilon \end{bmatrix} = P_v(\theta) \begin{bmatrix} V_a(t) \\ V_b(t) \\ V_c(t) \end{bmatrix}, P_v(\theta) = \frac{2}{3} \begin{bmatrix} sin(\theta + \frac{1}{2}\pi) & sin(\theta - \frac{1}{6}\pi) & -sin(\theta + \frac{1}{6}\pi) \\ -sin(\theta) & sin(\theta + \frac{1}{3}\pi) & sin(\theta - \frac{1}{3}\pi) \\ 0.5 & 0.5 & 0.5 \end{bmatrix}. \tag{5.8}$$

As (5.7) and (5.8) shown, the sinusoidal matrix is in terms of the angle θ between the a and q axes, which means in the each time step, the matrix need to be recalculated. The difficulty

of the park transformation function is that directly arithmetic calculation of sinusoidal values is unfeasible in hardware implementation. Therefore, we store sinusoidal values of angles ranging from 0 to $pi/2$ in memory in advance, then rely on the periodicity and symmetry of sine to map the original angle into the corresponding sinusoidal value directly. The block diagram is shown in 5.19 where matrix building block is a combination logic circuit for reverting the 3×3 matrix back.

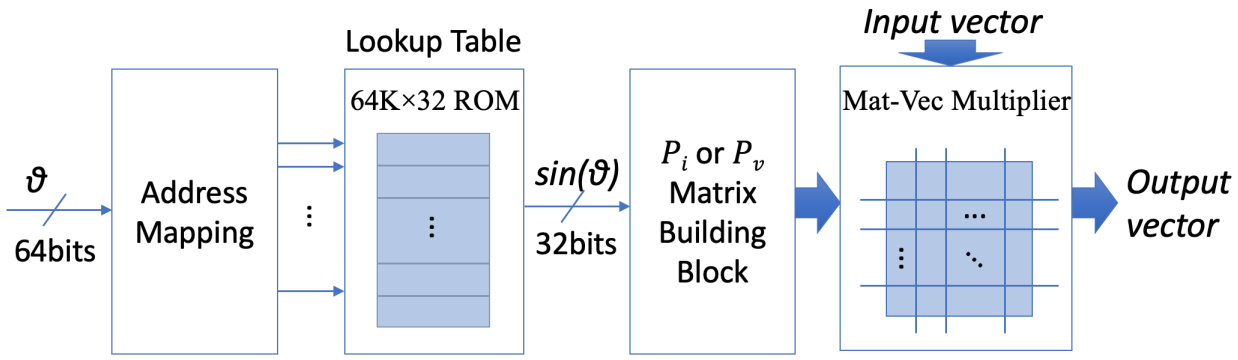


Figure 5.19: Circuit for park and inverse park transformation.

5.8.2 Network Solver

The network component consists of two sub-modules. One is to solve the network equation, the other one is to update history term of bus current used by network solution.

5.8.2.1 Network Equation Solver

As we discussed in Chapter 3, the network equation (3.3) is solved through a fast matrix-vector multiplication Algorithm 3 based on the hierarchical low-rank approximation of G^{-1}. According to Algorithm 3, the output **v** is calculated in two forms (3.12) and (3.13).

1. The matrix size involved in (3.12) is relative small which bounded by r_{th}, therefore, a general parallel architecture for matrix-vector multiplication is developed in Fig. 5.20.

2. The matrix size involved in (3.13) is relative large which depends on the size of power system. Therefore, a specific parallel architecture for fast matrix-vector multiplication is

needed. To reduce the computation, the σ in (3.13) is multiplied with v in advance, named the result as $v^{'}$, so that (3.13) is revised as (5.9). Consider about the trade-off between hardware resources and latency, we design a shift register to store $v^{'}$ and u. The detailed architecture is presented in Fig. 5.21.

$$\mathbf{v} = \sum_{i=1}^{r} u_i (v_i^{'} * \mathbf{i}). \tag{5.9}$$

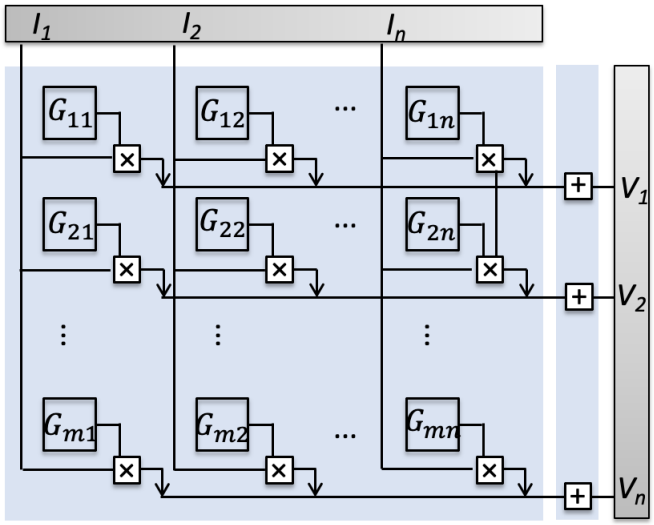


Figure 5.20: Architecture for general matrix-vector multiplication.

During the simulation, the data stored in v_i and u_i to shift from one location to the next. By connecting the last register back to the first, the data cycle within the shifters for extended periods. Therefore, the computation of the final result will cost r clock cycles.

5.8.2.2 Parallelism of Fast Network Solution

The low-rank approximation approach partitions the whole network hierarchically into $O(\log N)$ levels, where N is the number of buses. At each level, the computation involves mat-vec multiplications of total time $O(Nr_{\text{th}})$, where r_{th} is the user-defined rank threshold. Therefore, with N

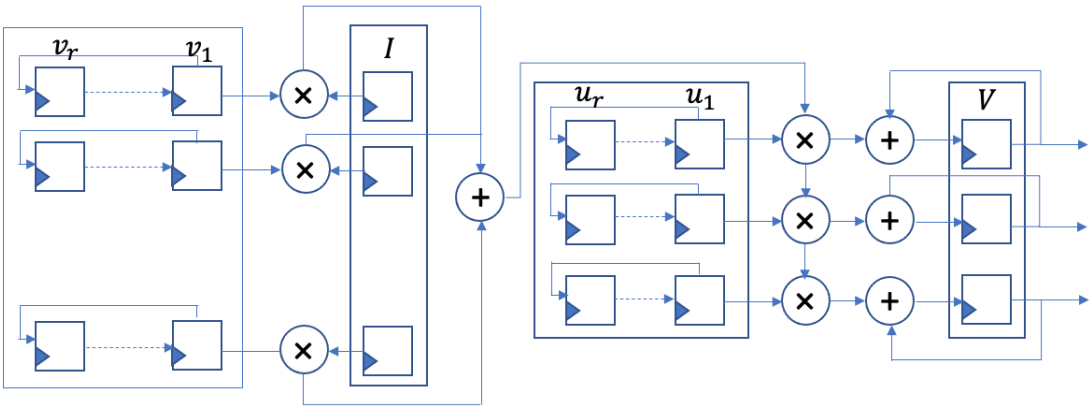


Figure 5.21: Circuit for "Network" module based on low-rank approximation approach.

processor units in parallel, the computation can be reduced to $O(r_{\text{th}})$. Since r_{th} is a constant, the proposed approach guarantees a network solution time of $O(\log N)$ using parallel techniques. This provides a significant advantage over LU based approaches.

5.8.2.3 Update Historical Bus Current

In the network equation, the bus current consists of two components. One component is coming from buses connected to the generator which is solved in the generator part. The other component is coming from history term of buses connected to network branches. The updating of history term of bus current I_{his} is computed by (5.10). The corresponding architecture is shown in Fig. 5.22. The designed circuit is a sequential circuit which takes one clock cycles to update the output. Registers are used to store history terms. Rest are 4 adders and 2 multipliers.

$$\begin{aligned} I_{lhis}(t+1) &= c_1(V_k - V_m) - c_2 I_{lhis}(t), \\ I^k_{his}(t+1) &= I^k_{his}(t) - I_{lhis}(t+1), \\ I^m_{his}(t+1) &= I^m_{his}(t) + I_{lhis}(t+1), \end{aligned} \tag{5.10}$$

where I_{lhis} is the branch current between bus k and bus m, I^k_{his} and I^m_{his} represent the history current on bus k and bus m, respectively.

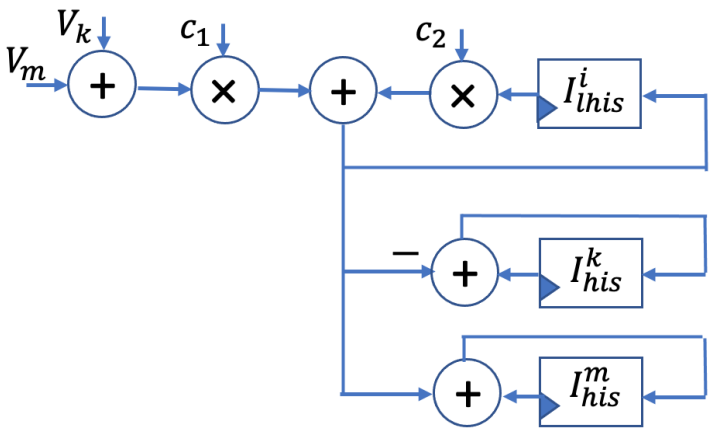


Figure 5.22: Circuit for "Update I_{his}" module.

5.8.3 Control Element Solver

Traditionally, control elements are modeled using control block diagrams in a frequency domain. Some control elements are also modeled as an actual electrical circuit for analysis of its performance. However, these approaches are all hard to map into the hardware-accelerated methods solvers such as FPGA-based [42]. This section will introduce a parallel VLSI approach to implement control elements.

In power system, most of control elements share similar model representations which contain magnitude exaction, transfer function, limiter, etc. Therefore, we create a custom component library of basic design modules making the design of varies types of control elements more flexible. The library is shown in Fig. 5.23. Since the exciter, PSS and governor share similar structure, we take implementation of exciter as an example.

5.8.3.1 Magnitude Extraction

The exciter makes use of the magnitude of terminal bus voltage to provide field voltage e_{fd}. Therefore, we need the magnitude extraction module to extract the voltage magnitude. To avoid the high frequency noises from varies sources, a digital low pass finite impulse response (FIR) filter is added at the beginning to make sure we catch the magnitude of voltage with 60HZ. After the original input is cleaned, magnitude is computed by magnitude prediction. The magnitude

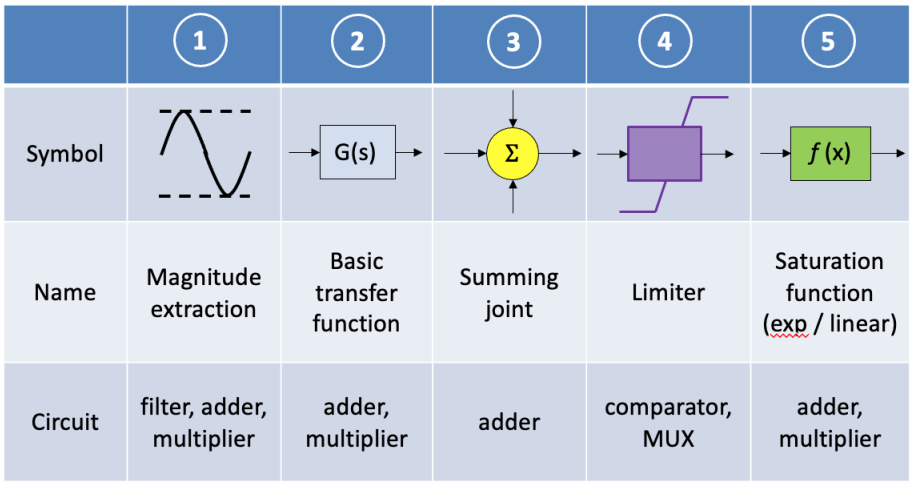


Figure 5.23: Component library of a transfer function.

extraction diagram is described in Fig. 5.24.

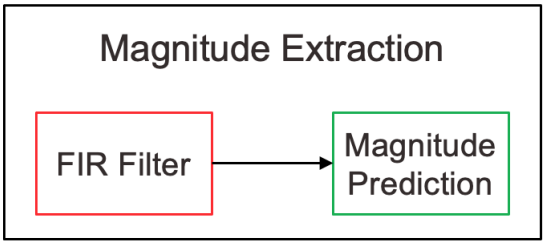


Figure 5.24: Magnitude extraction design.

FIR filter The FIR filter structure is shown in Fig. 5.25. In the Fig. 5.25, the filter is built by adders and a shifter. We store the initial 128 values of inputs first, then put the sum of the 128 values into a register. When a new value comes, update the sum register by adding the new value and subtract the oldest value, so that the high frequency component will be eliminated.

Magnitude prediction The magnitude prediction uses linear extrapolation approach to predict the magnitude. The pseudo algorithm is described below.

1. Locate first two absolute maximum points (P_1, t_1) and $(P_2^{'}, t_2)$ of the input signal, where $P_2^{'}$

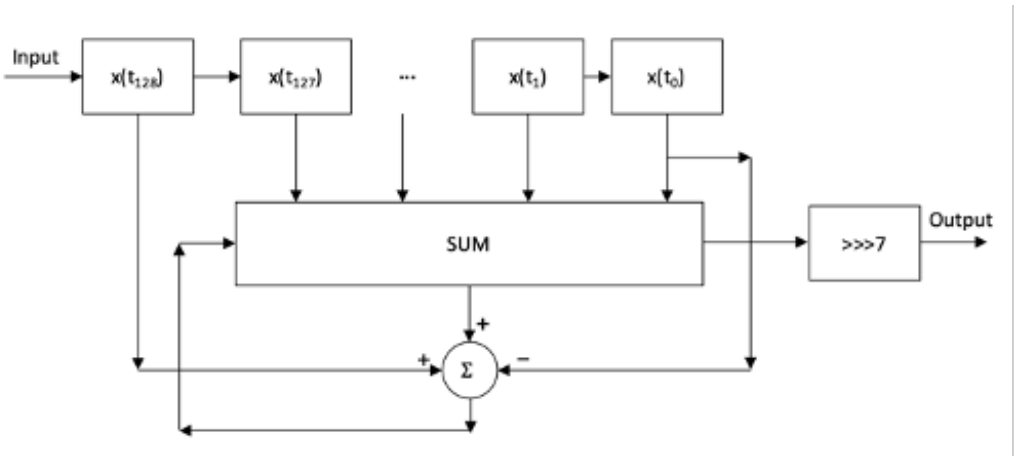


Figure 5.25: Hardware design of the FIR filter.

is the mirror of P_2.

2. Use linear extrapolation (5.11) to predict the magnitude

$$M(t) = \frac{P_2^{'} - P_1}{\Delta t}(t - t_2) + P_2^{'}. \tag{5.11}$$

3. When reach the next absolute maximum point (P_3, t_3), update (P_1, t_1) by ($P_2^{'}$, t_2) and ($P_2^{'}$, t_2) by (P_3, t_3), then repeat.

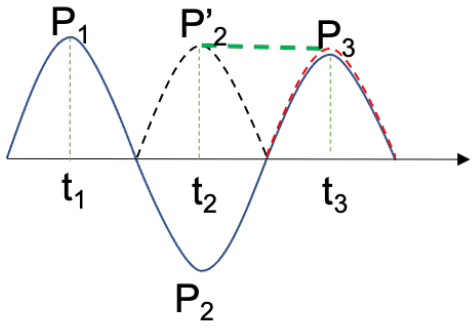


Figure 5.26: Magnitude prediction algorithm.

5.8.3.2 Transfer Function

A transfer function is defined as the ratio of the output of a system to the input of a system in the frequency domain. In the exciter or other control elements, multiple transfer functions are connected in series or in parallel to express the relationship between inputs and outputs. Following the principle of ASIC, we break down the entire control system from the input to the output into multiple individual transfer functions, then category the transfer function based on their formulation. Generally, transfer functions used in excitation system are integrator blocks, first order lag blocks, derivative blocks and lead-lag blocks. These blocks share one similar function formula shown Fig. 5.27. Given different coefficient, the formula can map into different functions.

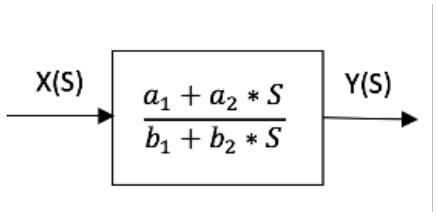


Figure 5.27: Standard one order transfer function.

To implement the transfer function in the time domain, we discretize the transfer function according to the Euler rule. The relationship between x and y in the time domain is shown in (5.12).

$$y(t + \Delta t) = \frac{b_2 - b_1 \Delta t}{b_2} y(t) + \frac{a_2}{b_2} x(t + \Delta t) + \frac{-a_2 + a_1 \Delta t}{b_2} x(t). \tag{5.12}$$

The implementation of exciters uses a parallel scheme where all state variables are updated concurrently using previous and current state variables, while in the traditional scheme state variables are updated sequentially. Instead of using a trapezoidal rule of integration [7] which needs multiple clock cycles, each transfer function block update is done in only one clock cycle. The implementation is shown in Fig. 5.28. In the figure, all coefficients are constants fixed before

simulation.

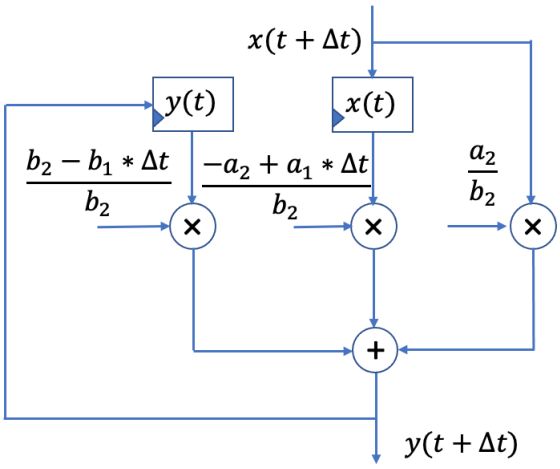


Figure 5.28: Hardware design of a transfer function.

5.8.3.3 Limiter

The limiter is to limit input values in a specific range. In the design, we use digital comparators to implement it. If the input value is larger than the pre-set upper limitation, output the upper value, else if the input value is less than pre-set lower limitation, output the lower value, else directly output the input value.

5.8.3.4 Saturation Function

In our design, exponential approach is used to represent the saturation, which is quite common adopted. The function is shown in (5.13).

$$\begin{aligned} S_E &= Me^{Ne_{fd}}, \\ M &= \frac{S_{e1}}{4log(S_{e1}/S_{e2})}, \\ N &= \frac{4log(S_{e1}/S_{e2})}{EV_{set}}. \end{aligned} \tag{5.13}$$

We design a incremental method to update S_E, which means we only need to calculate the difference part between current and previous value.

$$\begin{aligned} S_E(t) &= Me^{N(e_{fd}+\Delta e_{fd})} \\ &= S_E(t-1)e^{N\Delta e_{fd}}. \end{aligned} \tag{5.14}$$

Due to Δe_{fd} is close to zero and N is less than 1, we use Taylor series to approximate the exponential function,

$$e^{N\Delta e_{fd}} \approx 1 + N\Delta e_{fd}. \tag{5.15}$$

5.9 Logical and Layout Synthesis

A 2-bus system EMT simulation is designed by the full-cutom ASIC approach with TSMC $180nm$ technology. The generator and exciter modules are processing in parallel, which take 6 clock cycles in total. The network solution takes 2 clock cycles. Therefore, the total clock cycle for each iteration is 8. Through timing analysis, the longest delay is $20ns$ and therefore, the circuit is guaranteed to work at a clock rate of 50 MHz. Since VLSI circuit has not been fabricated, the output values are generated from logic simulation, and the delay is measured using timing simulation software.

The silicon layout of the system is shown in Fig. 5.29. The entire circuit size is $34.76mm^2$. The number of standard cells used in the design is 57968. Since the technology is quite old, if we use most advanced technology such as $5nm$ process, the circuit size will be reduced to $0.03mm$. Therefore, a $88mm^2$ chip, the size of Apple A14 used in iPhone12, will be able to contain 5870 generators.

More optimal and large-scale system design will be exploited and collaborated with Dr. Shi's master student Naga Shiva Sai Pavan Kumar Devarasetti.

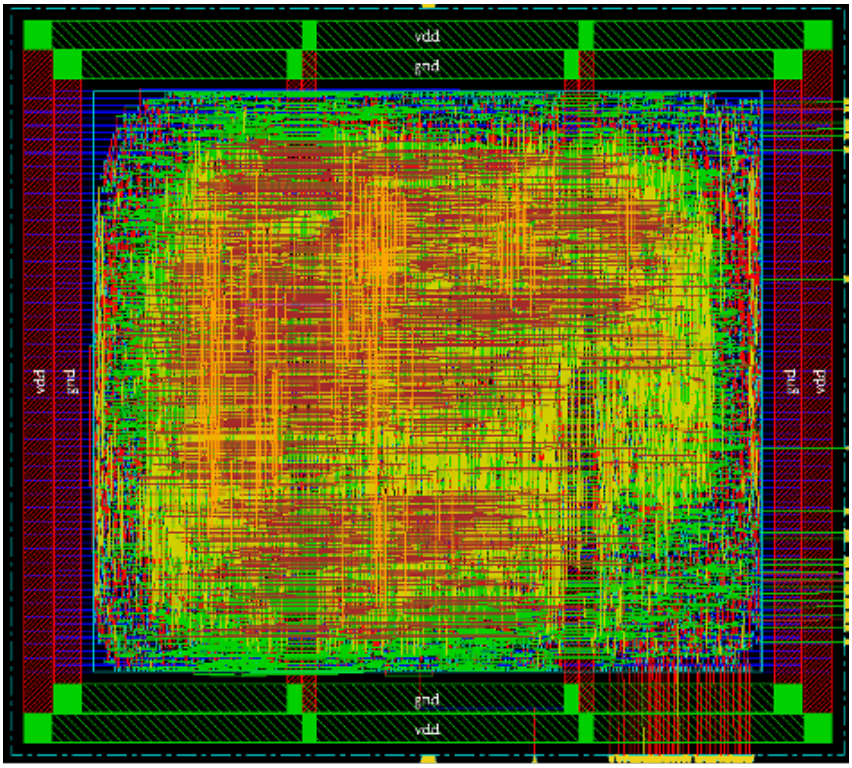


Figure 5.29: Layout of 2-bus system using a 180nm process.

5.10 Simulation Result

Let the same system simulation implemented in MATLAB be an reference. The comparison between ASIC and reference are presented in the following. The initial values of all variables are set to be 0 and the time step is fixed at $1.43\mu s$.

The results compared with MATLAB is shown in Fig. 5.30. The corresponding error is shown in Fig. 5.31. Fig. 5.31 shows that the phase A current of the generator computed by our VLSI circuit matches very well with that computed by MATLAB. The the maximum relative error is less than 0.4%.

5.11 Conclusion

In this chapter, a customized and parallel ASIC design for EMT simulation is developed. The implementation covers the entire power system including generator, exciter, and network. A simple system with 2 buses is implemented as an example. A fixed time-step approach is utilized.

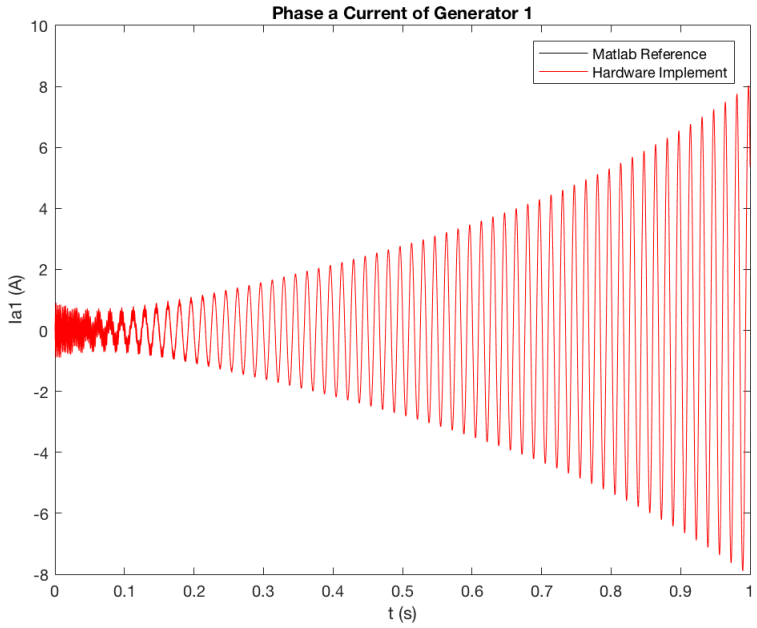


Figure 5.30: Comparison of phase A current for the generator.

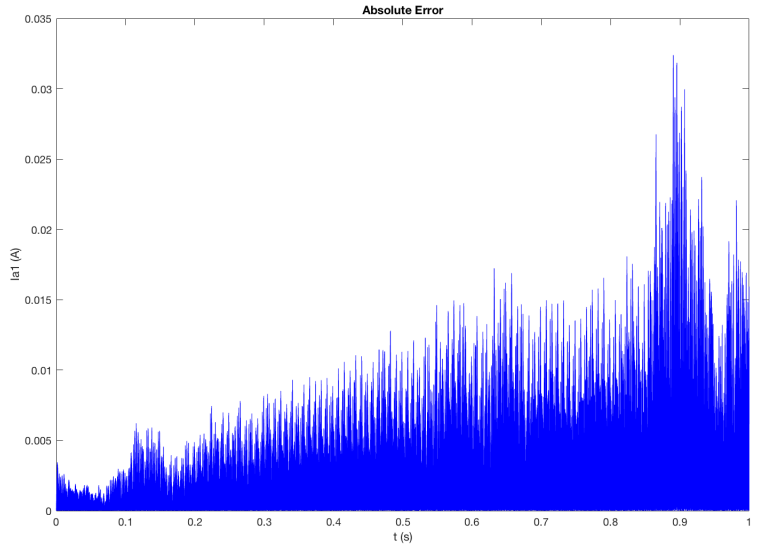


Figure 5.31: Relative error of phase A current for the generator.

Compared with MATLAB, our proposed simulator is highly accurate, and the error is within 0.4%. The proposed ASIC is implemented using TSMC $180nm$ process technology. Although a large

amount of computation is packed into each clock cycle, the clock can still reach 50MHz without timing error, due to our optimal design. Since each time step of $1.43\mu s$ requires 8 clock cycles, our circuit can simulate one time-step of the EMTP system in $160ns$, which is about 9 times faster than real-time.

6. CONCLUSIONS

In this dissertation, a fast EMT simulation solver based on a series of highly parallelized algorithms is presented which is enable to implement on the advanced high-performance hardware systems such as FPGA, GPUs, Clouds, full-custom ASIC, etc. To fully exploited the benefit of algorithms, we boost the performance by full-custom ASIC. The speedup techniques we proposed are mainly in three aspects.

First, we propose a highly parallel algorithm for solving network equations. In particular, we design a hierarchical truncated SVD to do the low-rank approximation in parallel, while traditional approaches are mostly based on LU factorization. Due to the inherently sequential dependency, traditional approaches are inefficient for hardware implementation. The LU factorization needs forward and backward substitution to solve the variables in each iteration. In contrast, the low-rank approximation approach we propose decomposes the network into multiple small groups where each group contains only one or two buses in the pre-step, and solves variables in these groups in parallel in each iteration. Analysis of the performance and accuracy are presented. Case studies are performed on small- and large-scale systems. It is observed that the proposed approach can speed up the network solution by up to 2.8 times compared to the sparse LU factorization based direct solver even in series computation and shows great scalability. In addition, since the proposed method is highly parallelizable, an additional speedup can be further achieved by using parallel computing.

Second, we break down the bottleneck of computing sparse matrix inversion in the inverse-based network solver by an efficient hierarchical approximation of computing and storage G^{-1} approach. Further, a fast G^{-1} modification approach is presented for contingency analysis. The performance is demonstrated by the EMT simulation on 179-bus based small- and large-scale systems. The experiments show that the proposed approaches exhibit great advantage in speeding up the network solution with high accuracy.

At last, a full-custom ASIC architecture with a massively parallelizable scheme for EMT sim-

ulation of an interconnected power system is developed. In the design, all state variables of generators and control elements are updated concurrently and state variables of multiple well-separated groups of buses are updated concurrently in network due to the hierarchical low-rank approximation scheme so that the speed of large-scale system EMT simulation is efficiently increased. A two-bus system is implemented using TSMC $180nm$ process technology. Although a large amount of computation is packed into each clock cycle, the clock can still reach 50MHz without timing error, due to our optimal design. Since each time step of $1.43\mu s$ requires 8 clock cycles, our circuit can simulate one time-step of the EMT system in $160ns$, which is about 9 times faster than real time. Nevertheless, $180nm$ is almost two decades old. Had we used today's best technology, our approach may lead to another order of magnitude speed improvement, and allow one single silicon chip to simulate 5870 generators.

REFERENCES

[1] R. Huang, S. Jin, Y. Chen, R. Diao, B. Palmer, Q. Huang, and Z. Huang, "Faster than real-time dynamic simulation for large-size power system with detailed dynamic models using high-performance computing platform," in *2017 IEEE Power Energy Society General Meeting*, pp. 1–5, 2017.

[2] X. Zhang, A. J. Flueck, and S. Abhyankar, "Implicitly coupled electromechanical and electromagnetic transient analysis using a frequency-dependent network equivalent," *IEEE Transactions on Power Delivery*, vol. 32, pp. 1262–1269, Jun. 2017.

[3] A. J. Flueck, "High-fidelity, faster than real-time dynamics simulation," in *2014 IEEE PES General Meeting | Conference Exposition*, pp. 1–1, 2014.

[4] J. Mahseredjian, V. Dinavahi, and J. A. Martinez, "Simulation tools for electromagnetic transients in power systems: Overview and challenges," *IEEE Transactions on Power Delivery*, vol. 24, no. 3, pp. 1657–1669, 2009.

[5] P. Le-Huy, M. Woodacre, S. Guérette, and E. Lemieux, "Massively parallel real-time simulation of very-large-scale power systems," 2017.

[6] H. W. Dommel, "Digital computer solution of electromagnetic transients in single-and multiphase networks," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-88, no. 4, pp. 388–399, 1969.

[7] N. Watson and J. Arrillaga, "Power systems electromagnetic transients simulation," *U.K., London: Inst. Electr. Eng*, pp. 193–198, 2003.

[8] A. Ametani, *Numerical Analysis of Power System Transients and Dynamics*. The Institution of Engineering and Technology, Jan. 2015.

[9] Q. Huang and V. Vittal, "Application of electromagnetic transient-transient stability hybrid simulation to fidvr study," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 2634–

2646, 2016.

[10] L. Wang, J. Jatskevich, V. Dinavahi, H. Dommel, J. Martinez, K. Strunz, M. Rioual, G. Chang, and R. Iravani, "Methods of interfacing rotating machine models in transient simulation programs," *IEEE Trans. on Power Delivery*, vol. 25, pp. 891–903, Apr. 2010.

[11] Y. Chen and V. Dinavahi, "FPGA-based real-time EMTP," *IEEE Transactions on Power Delivery*, vol. 24, no. 2, pp. 892–902, 2009.

[12] Y. Song *et al.*, "Fully GPU-based electromagnetic transient simulation considering large-scale control systems for system-level studies," *IET Generation, Transmission Distribution*, vol. 11, no. 11, pp. 2840–2851, 2017.

[13] Manitoba HVDC Research Centre, "PSCAD homepage." `https://www.pscad.com/software/pscad/overview`.

[14] Powersys Solutions Software and Services, "EMT-RV: The reference for power systems transients." `http://EMT-software.com/`.

[15] RTDS Technologies Inc, "Real Time Digital Power System Simulator." `https://www.rtds.com/`.

[16] Opal-RT Technologies, "Real-Time Simulation Real-Time Solutions OPAL-RT." `https://www.opal-rt.com/`.

[17] V. Spitsa, R. Salcedo, X. Ran, J. F. Martinez, R. E. Uosef, F. de Leon, D. Czarkowski, and Z. Zabar, "Three–phase time–domain simulation of very large distribution networks," *IEEE Transactions on Power Delivery*, vol. 27, no. 2, pp. 677–687, 2012.

[18] L. Gérin-Lajoie and J. Mahseredjian, "Simulation of an extra large network in EMTP: From electromagnetic to electromechanical transients," *Int. Conf. Power Syst. Transients*, Jun. 2019.

[19] Y. Chen, Y. Song, S. Huang, Z. Yu, and W. Wei, "Gpu-based techniques of parallel electromagnetic transient simulation for large-scale distribution network," *Dianli Xitong Zidonghua/Automation of Electric Power Systems*, vol. 41, pp. 82–88, Oct. 2017.

[20] A. Abusalah, O. Saad, J. Mahseredjian, U. Karaagac, L. Gerin-Lajoie, and I. Kocar, "CPU based parallel computation of electromagnetic transients for large power grids," *Electric Power Systems Research*, vol. 162, pp. 57–63, Sep. 2018.

[21] Z. Zhou and V. Dinavahi, "Fine-grained network decomposition for massively parallel electromagnetic transient simulation of large power systems," *IEEE Power and Energy Technology Systems Journal*, vol. 4, no. 3, pp. 51–64, 2017.

[22] P. Krause, O. Wasynczuk, S. Sudhoff, and S. Pekarek, *Analysis of Electric Machinery and Drive Systems*. John Wiley & Sons, 2013.

[23] PowerWorld Corporation, "Web help." `https://www.powerworld.com`.

[24] K. Takahashi, J. Fagan, and M. Chin, "Formation of a sparse bus impedance matrix and its application to short circuit study," *8th PICA Conference Proc.*, pp. 177–179, Jun. 1973.

[25] J. Mahseredjian, I. Kocar, and U. Karaagac, *Solution Techniques for Electromagnetic Transients in Power Systems*, ch. 2, pp. 9–38. John Wiley & Sons, 2015.

[26] Hydro-Quebec, "Simscape power systems user's guide 2018a," Mar. 2018.

[27] C. Dufour, H. Saad, J. Mahseredjian, and J. Belanger, "Custom-coded models in the state space nodal solver of artemis," 2013.

[28] G. Golub and C. V. Loan, *Matrix Computations*. Johns Hopkins University Press, 3 editions, 2012.

[29] S. Yan, V. Sarin, and W. Shi, "Sparse transformations and preconditioners for 3-D capacitance extraction," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 1420–1426, Sep. 2005.

[30] T. Athay, R. Podmore, and S. Virmani, "A practical method for the direct analysis of transient stability," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-98, no. 2, pp. 573–584, 1979.

[31] S. Maslennikov, B. Wang, Q. Zhang, a. Ma, a. Luo, a. Sun, and E. Litvinov, "A test cases library for methods locating the sources of sustained oscillations," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pp. 1–5, 2016.

[32] Y. Chen and V. Dinavahi, "Hardware emulation building blocks for real-time simulation of large-scale power grids," *IEEE Transactions on Industrial Informatics*, vol. 10, pp. 373–381, Feb. 2014.

[33] T. Davis, "Summary of available software for sparse direct methods," Jan. 2009.

[34] L. Greengard and W. D. Gropp, "A parallel version of the fast multipole method," *Computers & Mathematics with Applications*, vol. 20, no. 7, pp. 63–71, 1990.

[35] F. Milano, *Power System Modelling and Scripting*. U.K., London: Springer, 2010.

[36] K. Abdul-Rahman, J. Wu, E. Haq, F. Howell, X. Lin, and L. Wang, "Application of high performance computing in california ISO's on-line DSA system," in *2017 IEEE Power Energy Society General Meeting*, pp. 1–7, 2017.

[37] B. D. Kroposki, "Summarizing the technical challenges of high levels of inverter-based resources in power grids," May 2019.

[38] B. Wang, N. Duan, and K. Sun, "A time–power series-based semi-analytical approach for power system simulation," *IEEE Transactions on Power Systems*, vol. 34, no. 2, pp. 841–851, 2019.

[39] T. Davis and E. Natarajan, "(2010) algorithm 907: KLU, a direct sparse solver for circuit simulations problems," *ACM Transactions on Mathematical Software*, vol. 37, no. 3, p. 36:1–36:17, 2010.

[40] L. Zhang, B. Wang, L. Wu, D. Xie, P. R. Kumar, and W. Shi, "Fast electromagnetic transient simulation based on hierarchical low-rank approximation," in *IEEE ISGT*, pp. 1–5, 2019.

[41] L. Zhang, B. Wang, X. Zheng, W. Shi, P. R. Kumar, and L. Xie, "A hierarchical low-rank approximation based network solver for emt simulation," *IEEE Transactions on Power Delivery*, pp. 1–1, 2020.

[42] L. Zhang, M. Wu, Z. Li, P. R. Kumar, L. Xie, and W. Shi, "VLSI architecture for exciter, governor, and stabilizer in fast power system EMT simulation," in *IEEE TPEC*, pp. 1–6, 2018.