Parallel-in-Time-and-Space Data-Oriented Electromagnetic Transient and Dynamic Simulation of AC-DC Smart Grids

by

Tianshi Cheng

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

Doctor of Philosophy
in
Energy Systems

Department of Electrical and Computer Engineering

University of Alberta

Abstract

This thesis pioneers new PiT and PiS acceleration techniques in the transient simulation of smart grids, introducing a series of algorithms, software technologies, and heterogeneous parallel computing practices that are the first of their kind in this field. It begins with an exploration of new PiT algorithms, particularly focusing on the application of the Parareal algorithm in accelerating AC-DC power grid simulations. The algorithm study addresses a series of theoretical and engineering challenges associated with PiT algorithms, demonstrating their value through the implementation of a massively parallel PiT+PiS algorithm on a heterogeneous CPU-GPU architecture for EMT-TS co-simulation scenarios.

Meanwhile, the complexity of heterogeneous PiT+PiS algorithm implementation has brought new challenges to traditional software design philosophies. Further exploration revealed the advantages of new data-oriented software architectures over traditional object-oriented architecture to address the issues for exploring new applications and algorithms. In response, the first data-oriented cyber-physical power system simulation platform: ECS-Grid is developed based on the novel ECS architecture to address the demanding cyber-physical co-simulation issues for smart grids. The ECS-Grid provided better flexibility and performance compared to traditional solutions in complex cyber security scenarios running on distributed real-time hardware. An interdisciplinary digital-twin simulation solution leveraging LEO satellite networks is further developed to demonstrate the flexibility and superiority of the

new data-oriented ECS-Grid platform. The studies have shown that the data-oriented ECS paradigm can be a promising choice for developing massively parallel heterogeneous PiT and PiS simulation programs of smart grids.

To address the computational challenges posed by the massive integration of nonlinear models of renewable energy sources in smart grids, a new PiS acceleration method based on machine learning and neural network technologies is proposed. This novel method uses traditional nonlinear simulation models and Monte Carlo simulations to generate reliable training data for data-driven machine learning models. With the powerful ECS-based architecture, the optimal batching parallel processing on GPUs is achieved by modular designs, which fully leverage the technical advantages of the ECS-Grid platform. The proposed new machine-learning-reinforced parallel acceleration method has shown significant speed-up compared to traditional sequential nonlinear circuit simulation and can better utilize the heterogeneous hardware resource.

Overall, the first-of-their-kind contributions of this thesis not only advance the theory and practical application of smart grid simulations but also pave the way for future innovations in power system transient analysis and applications.

Preface

The material presented in this thesis is based on original work by Tianshi Cheng. As detailed in the following, material from some chapters of this thesis has been published in conference proceedings and as journal articles under the supervision of Dr. Venkata Dinavahi in concept formation and by providing comments and corrections to the article manuscript.

Chapter 2 includes the results from the following papers:

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

- **T. Cheng**, N. Lin, T. Liang, V. Dinavahi, "Parallel-in-time-and-space electromagnetic transient simulation of multi-terminal DC grids with device-level switch modelling," *IET Generation, Transmission & Distribution*. Vol. 16, pp. 149-162, 2022.

Chapter 3 includes the results from the following papers:

- **T. Cheng**, N. Lin and V. Dinavahi, "Hybrid parallel-in-time-and-space transient stability simulation of large-scale AC/DC grids," *IEEE Transactions on Power Systems*, vol. 37, no. 6, pp. 4709-4719, Nov. 2022.

Chapter 4 includes the results from the following papers:

- **T. Cheng**, T. Duan and V. Dinavahi, "ECS-Grid: data-oiented real-time simulation platform for cyber-physical power systems," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 11, pp. 11128-11138, Nov. 2023.

Chapter 5 includes the results from the following papers:

- **T. Cheng**, T. Duan and V. Dinavahi, "Real-time cyber-physical digital twin for low earth orbit satellite constellation network enhanced wide-area power grid," submitted to *IEEE Open Journal of the Industrial Electronics Society*, pp. 1-9, 2024.

Chapter 6 includes the results from the following papers:

- **T. Cheng**, R. Chen, N. Lin, T. Liang and V. Dinavahi, "Machine-learning-reinforced massively parallel transient simulation for large-scale renewable-energy-integrated power systems," *IEEE Transactions on Power Systems*, pp. 1-12, 2024, doi: 10.1109/TPWRS.2024.3409729.

To my Parents

For their unconditional support and love.

Acknowledgements

I'm truly grateful to Dr. Venkata Dinavahi, the supervisor of my Ph.D. journey at the University of Alberta. It's not just the wisdom and the research insights he generously shared, but also his infectious enthusiasm, unwavering dedication, and the spark of positivity he carries. These gifts inspired me to face and conquer the challenges with a brave heart. and I'm deeply grateful to all my colleagues and friends for their invaluable advice and assistance. I also want to extend my heartfelt gratitude to my parents for their endless love, unwavering support, and constant encouragement.

This thesis work was supported by NSERC, the University of Alberta, Mitacs, and RTDS Technologies Inc. I greatly appreciate their financial support.

Contents

List of Tables

List of Figures

List of Acronyms

AC	Alternative Current
AI	Artificial intelligence
ANN	Artificial Neural Network
AOS	Acquisition of the Satellite
API	Application Interface
CPPS	Cyber-Physical Power System
CPU	Central Processing Unit
DAE	Differential-Algebraic Equation
DAG	Directed Acyclic Graph
DC	Direct Current
DDE	Delay Differential Equation
DFIG	Doubly-Fed Induction Generator
ECEF	Earth-centered,Earth-Fixed
ECI	Earth-centered Inertial
ECS	Entity-Component-System
EMT	Electromagnetic Transient
GEO	Geosynchronous Earth Orbit
GMST	Greenwich Mean Sidereal Time
GPU	Graphical Processing Unit
GRU	Gated Recurrent Unit
HBSM	Half-Bridge SubModule
HVDC	High-Voltage Direct Current
IGBT	Insulated Gate Bipolar Transistor
JD	Julian Day
LEO	Low Earth Orbit
LOS	Loss of the Satellite
MLP	Multi-Layer Perceptron

MMC	Modular Multilevel Converter
MPSoC	Multiprocessor Systems-on-Chips
MSE	Mean Square Error
MTDC	Multi-Terminal Direct Current
NLM	Nearest-Level Modulation
NPU	Neural Processing Unit
ODE	Ordinary Differential Equation
OOP	Object-Oriented Programming
PMU	Phasor Measurement Unit
PV	Photovoltaic
PiS	Parallel-in-Space
PiT	Parallel-in-Time
RES	Renewable Energy System
RNN	Recurrent Neural Network
SDN	Software-Defined Network
SM	Sub-Module
TEME	True-Equator-Mean-Equinox
TLE	Two-Line Element
TLM	Transmission Line Model
TS	Transient Stability
UTC	Coordinated Universal Time
VSC	Voltage-Source Converter
WAMS	Wide Area Measurement System
WGS84	World Geodetic System 1984

Chapter 1

Introduction

The smart grid is becoming a multidirectional super highway. Billions of devices ranging from giant nuclear power plants to tiny household energy storage are pushing and pulling energy to and from the grid all the time. Such a complex system cannot stand without reliable analytical data from various types of simulation. The electromagnetic transient (EMT) simulation is paramount for every step of design, testing, commissioning, operation, control, and protection in smart grids [1]. The EMT simulation is very data-intensive and compute-intensive, while the smart grid always demands faster simulation speed and larger system scales. A widely used solution is to utilize parallel computing techniques to accelerate the simulation. The traditional methods are based on parallel-in-space (PiS) algorithms which partition a large system into many small subsystems and solve them concurrently. However, PiS methods are not the only choices. This thesis demonstrates brand-new parallel-in-time (PiT) methods to accelerate the EMT simulation of AC and DC power grids. The PiT algorithm: Parareal is fully examined in various simulation scenarios. While PiT methods provide new opportunities for AC and DC power electronic EMT simulation,parallel-in-time-and-space methods, which combine both advantages of PiT and PiS methods are proposed to solve large-scale EMT and transient stability co-simulation. The heterogeneous CPU-GPU design provides excellent parallel efficiency and is the most promising way to further improve the performance of EMT programs. Through the research of PiT and PiT+PiS algorithms, the traditional and dominant object-oriented pro-

gramming (OOP) paradigms have become an obstruction to future progressions. The PiT algorithm is quite complicated, making it challenging to integrate with traditional simulation software. The increasing complexity of the PiT+PiS algorithm and heterogeneous computing based on the OOP design is now an obstruction for practical applications. Similar challenges also arise for interdisciplinary simulation needs, such as cyber-physical simulations, which are playing an increasingly significant role in modern smart grids.

Data-oriented entity-component-system (ECS) paradigms are promising for the future foundation of new algorithm development and smart grid simulation applications. To explore the potentials of the data-oriented ECS design, the ECS-Grid: a data-oriented real-time EMT simulation platform for cyber-physical power system (CPPS) co-simulation is proposed. It demonstrates high flexibility, scalability, and optimal performance to investigate cyber-physical scenarios in smart grids. Furthermore, machine-learning neural network models for renewable energy systems (RESs) are integrated into the data-oriented platform. Based on the ECS-Grid platform, a practical and efficient method is proposed to accelerate the RES neural network models on heterogeneous CPU-GPU architecture.

Overall, this research has encompassed algorithm exploration, software implementation, and practical applications across the spectrum of smart AC-DC grid simulation, to introduce comprehensive new data-oriented approaches to inspire future innovations in power system transient simulation.

1.1 Literature Review

1.1.1 Parallel-in-Time and Parallel-in-Space Methods

The parallel-in-space (PiS) methods can be categorized into:

1. Non-Iterative methods: These methods try to directly partition a system from its physical structure. For example, the traveling-wave line model naturally decoupled subsystems due to the delay of wave transmission [2]. For many scenarios, a single-step delay can be added to decouple power electronic equipment and parallelize the computing [3], [4].

Another way is to optimize the matrix algorithms with SIMD instructions. Replacing matrix algorithms with SIMD-optimized basic linear algebra subprograms (BLAS) such as Intel® MKL and OpenBLAS can bring significant acceleration [5]. These acceleration libraries use many highly optimized kernel functions that can work efficiently on fixed-size partitions, and the large-scale problem can be divided and solved by SIMD-optimized kernels and thus achieve higher performance compared to plain BLAS implementation.

2. Iterative methods: these methods are more common in finite element analysis due to the huge problem size and nonlinearity. The domain decomposition methods such as Schwarz, Neumann-Neumann, and Optimized Schwarz are widely used to accelerate PDE and ODE problems in electrical engineering with parallel computing [6]–[8]. The iterative methods are often more indirect and have fewer constraints compared to non-iterative PiS methods. But they also bring convergence problems and may have lower efficiency compared to the direct non-iterative PiS methods.

For EMT simulation, the non-iterative PiS methods are the main streams for parallel computing. The PiS methods can be implemented on the graphic processing unit (GPU) to accelerate large-scale power grid simulations [9] or implemented on FPGAs to realize the real-time performance for the hardware-in-the-loop simulation of detail-modeled power electronic converters [10]–[13].

Multi-rate simulation methods are proposed to push the boundary further. In 1993, [14] proposed the early version of multi-rate EMT simulation. It aimed to explore With the rapid development of parallel hardware, recently multi-rate methods become more popular, especially for the co-simulation of heterogeneous system models [4], [15], [16] and hybrid CPU-GPU computation [17]. Although these computing methods achieved better speed-ups using multiple time steps, they are still based on physical topologies. Consequently, the efficiency is limited by the spatial partitions of systems.

Time-domain decomposition methods such as parallel-in-time (PiT) algo-

rithms provide a new perspective regarding parallel computing and have shown effectiveness in fields involving ordinary differential equations (ODEs) [18]. Although the PiT algorithms may use multiple time steps, which is similar to multi-rate methods, their theoretical foundations are different. The multi-rate methods are still considered PiS methods since the systems are often decomposed into different subsystems at different time steps. For PiT algorithms, the system is decomposed on multiple time grids, which means the system scale can be very small but still get considerable acceleration from PiT computing. A popular PiT algorithm is the Parareal algorithm, which solves initial value problems iteratively by using two ODE integration methods. It has become one of the most widely studied PiT integration methods [18]. It has also been proven to be able to solve semi-differential algebraic equation (DAE) or full-DAE problems in transient stability simulation [19], [20], and eddy current calculation [21].

1.1.2 Heterogeneous CPU-GPU PiT+PiS Simulation for TS-EMT Co-Simulation

Using a typical step size of a few milliseconds, the transient stability (TS) analysis plays an important role in the planning, design, and operation of a modern power grid from a system point of view. It, however, is a positive-sequence-based analytical method that naturally falls short of complicated electromagnetic transient (EMT) details of power electronic devices in the microsecond level or below. The TS-EMT co-simulation methodology properly features system-level and equipment-level power system phenomena is favored in hybrid AC-DC grid study. A consequently incurred rise of computational burden may extend the simulation duration, especially considering that a dramatic expansion of a future AC-DC power system as a result of incorporating more components is expected.

To handle the increasing scale and complexities, new acceleration techniques for TS and EMT simulation programs are desired. TS acceleration methods based on parallel processing algorithms for multi-core CPUs and many-core GPUs have shown a decent efficiency and have been well inves-

tigated in AC power grid studies [22]–[24] , and heterogeneous CPU-GPU computing architecture for AC-DC grid TS-EMT co-simulation has recently been proposed [25]–[27], while the threads concurrency of these methods is dominantly contributed by PiS strategies.

PiT solutions also exist for TS simulation. The very early version of PiT for solving TS problems was mainly based on Jacobi-decomposition proposed by La Scala [28], [29], which aimed to solve multiple continuous steps iteratively so that the parallelism was achieved by simultaneously solving multiple time-steps. In the 1990s, most results were obtained from the virtual parallel machine because of a scarcity of multi-core CPUs, which limited further explorations in this area. Different from La Scala's method, the Parareal algorithm can solve TS simulation problems by decomposing the initial value problem into many sub-intervals [19] and has a better efficiency compared to its predecessors [30]. These works mainly focused on PiT algorithms and potential comprehensive parallelism by considering PiS methods simultaneously is yet to be carried out. For example, a four times speedup was obtained in computing the IEEE 39-bus system [19] with 470 cores, and even if a huge number of cores were used to solve a large-scale power system, the parallel efficiency was below 20% [31], which is still not as satisfactory as PiS methods. Most threads are wasted from the results of these research works. The efficiency can be improved by combining both advantages of PiT and PiS methods, which is the parallel-in-time-and-space (PiT+PiS) solution. Meanwhile, the PiT+PiS method can better utilize the abundant parallel resource on heterogeneous CPU-GPU hardware, which can bring a better solution for TS-EMT co-simulation of large-scale systems.

1.1.3 Data-Oriented Simulation for Smart Grid

The power system EMT simulation programs such as PSCAD®/EMTDC [32] and EMTP [33], [34] are mainly developed before the 1990s. The foundation of these programs is Fortran subroutines so that the architecture design is fixed to a procedure-oriented programming pattern. In PSCAD®/EMTDC, PSCAD serves as a graphical front-end interface for users and EMTDC is the

Fortran backend to run the simulation program. A complex code generation system is used to hide all complexity to generate and compile the Fortran codes for simulation programs. It is also the same architecture for real-time simulators such as RTDS's NovaCor and OpalRT's RTLAB, which uses RSCAD® or MATLAB/Simulink as the front-end and generates the codes for the real-time simulator. The old procedural-oriented design and complex code generation system ensured a good performance as well as a flexible and easy-to-use frontend for users. However, the drawbacks are also obvious: 1. code generation is required before running any simulation, which brings additional complexities; 2. although generated codes can run very fast, the solvers and algorithms are fixed. Due to these limitations, it is hard to explore new algorithms and simulation methods on the old simulation architecture.

Due to the increasing complexities, The current simulation software architecture cannot satisfy the demands of developing new practical PiT and PiT+PiS algorithms on heterogeneous hardware. Meanwhile, smart grids are advancing towards enhanced intelligence and environmental sustainability, which brings more challenges to power system simulation. For instance, the emergence of cyber-physical power system simulations [35] reflects the growing demands for new interdisciplinary simulation applications. However, the current simulation solutions based on legacy system architectures [36]–[39] cannot address these challenges comprehensively and efficiently. Considering both academic and practical perspectives, recent advancements in network communication technologies [40], the rise of novel operational scenarios [41], [42], and the integration of renewable energy systems highlight the need for researchers to create new flexible and adaptable high-performance simulation architectures.

The emerging data-oriented Entity Component System (ECS) architecture offers promising solutions for next-generation simulation software. The ECS framework has been successfully implemented in large-scale gaming projects, including the renowned Minecraft [43], providing crucial technical support for efficiently simulating complex game scenarios in these vast real-time software projects [44]. Moreover, the ECS framework is gaining attention in

robotics simulation, signifying its growing influence and potential. Open Source Robotics Foundation has an open source software for general robot simulation: Gazebo [45]. It has been renovated with the ECS framework as the backbones to achieve generality and high performance [46].

ECS architecture optimizes data locality by separating data and logic. This separation results in improved performance, especially in handling large-scale data, as opposed to the traditional OOP approach, where the merging of data and logic can reduce data processing efficiency. Therefore, ECS architecture is more suitable for designing heterogeneous parallel computing algorithms.

Furthermore, the data-oriented programming in ECS facilitates the data component combinations instead of inheritance, aligning closely with engineering needs. In real-world scenarios, utility companies are eager to establish a data fusion platform, that can integrate various data sources and provide enhanced system analysis and decision-making capabilities [47]. The innate data fusion capability of ECS is highly beneficial for interdisciplinary simulation integration and handling complex simulation environments such as cyber-physical co-simulation.

The design of a data-oriented ECS framework is evolving rapidly. Currently, two primary types are emerging: the sparse set-based ECS [43] and the archetype-based ECS [44], [48]. Archetype-based ECS frameworks are better suited for high-performance power system simulations due to their optimal data layouts for parallel computing. Based on these benefits, the data-oriented ECS framework is promising for a new foundation to develop complex heterogeneous algorithms and implement new applications such as cyber-physical simulation for future smart grids.

1.1.4 AI-Enhanced Methods for Large-Scale RES Simulation

Nowadays, EMT simulation plays a significant role in analyzing power grids with RES integrations. However, the scale of RESs is much larger than traditional power systems. There are more than 300,000 PV panels in a 100MW solar power farm [49], while each module may have an impact on the entire

solar farm performance in partial shading scenarios [50], [51]. The same problem also exists for battery groups where the battery management system needs to take care of inconsistencies within the series battery array to maintain the optimal performance [52]. Traditional parallel computing techniques and algorithms cannot efficiently handle such large-scale nonlinear systems. However, AI machine-learning technologies may provide a better solution.

The AI machine-learning technologies have achieved significant successes in various complex tasks such as image synthesis [53], natural language processing [54] and weather forecasting [55]. The fundamental of these AI technologies is the artificial neural network which is a machine learning technology that can be conceptualized as a mathematical approach to multivariate nonlinear regression [56]. In smart grid research, these technologies have attracted significant attention from power system researchers as well [57]–[59]. While machine learning technologies are popular in long-term and steady-state power system analysis [60]–[62], the utilization of machine learning technologies is just the beginning for power system electromagnetic transient (EMT) simulation. Some research works such as [63]–[65] have demonstrated the advantages and benefits of using ANN and RNN technologies to accelerate real-time EMT models on FPGA. However these works are early explorations aimed to deal with traditional components in power systems for specific scenarios, and such research hasn't been extended to RES modeling. Moreover, the integration of the ANN models into conventional EMT solvers is important for practical large-scale simulation applications but it was not comprehensively explained in previous research works.

The traditional nonlinear algorithm for RESs was implemented on GPUs before [66], [67], but the GPUs lack many instructions and have wrap divergence problems, leading to a low parallel efficiency. Due to the ANN computations being generic matrix operations, it can fully utilize the computing power of massively parallel hardware such as a Graphical Processing Unit (GPU). Meanwhile, modern CPUs and SoCs are tending to equip the NPUs specialized for ANN computing. These new NPUs cannot be used by traditional algorithms so the AI-enhanced EMT simulation is desired to take advantage

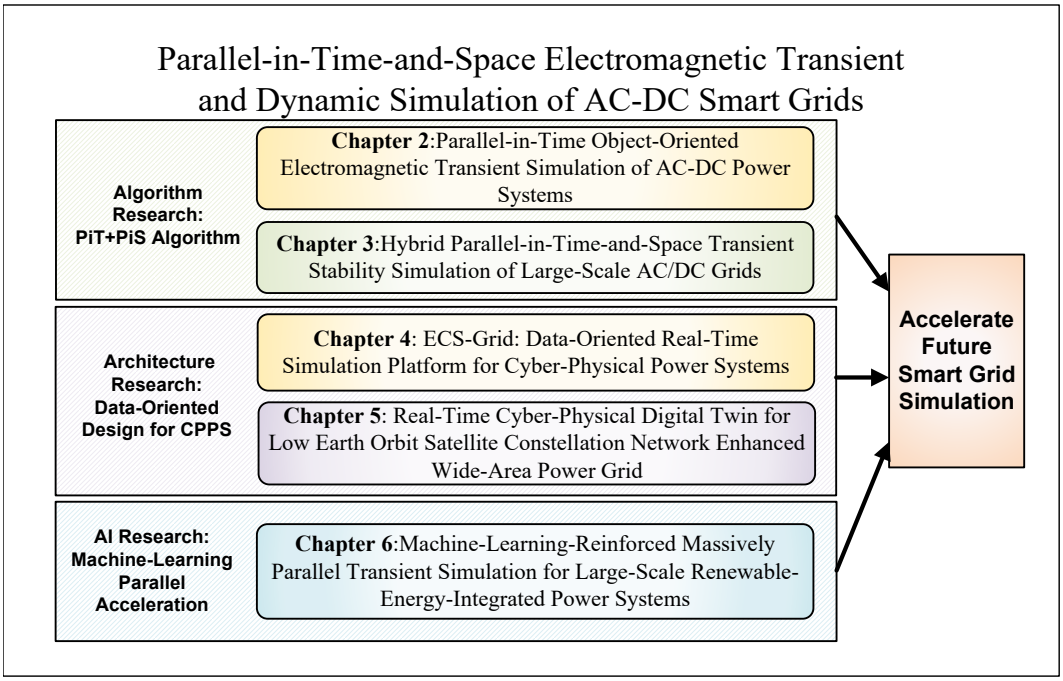

Figure 1.1: Summary of thesis research contributions.

of the modern hardware. Therefore, the AI-enhanced EMT simulation methods can inspire new PiS and PiT simulation algorithms in the new generation of heterogeneous hardware.

1.2 Summary of Contributions

As shown in Fig. 1.1, the major contributions of this thesis are summarized as follows:

- **PiT Algorithm Implementation for AC Systems with Traveling-Wave Line Models**

  A component-based simulation class architecture is proposed to handle different models in power systems, which delivers high flexibility and scalability to implement the system-level PiT algorithm. Major challenges to handle the DDEs brought by transmission line models in the PiT algorithm are analyzed and a modified model implementation is proposed to solve the problem. Using the proposed circuit solver class as basic workers, the PiT algorithm based on the Parareal is implemented using

object-oriented C++. The performance tests of IEEE AC grid cases show accurate results compared to the sequential program, while better parallel speed-up and efficiency than the MKL parallel LU implementation are obtained, showing the great potential of accelerating power system EMT simulation. The performance test also shows the system's time-domain characteristics determine the speed-up of the Parareal algorithm, especially the transmission line delay in power system EMT simulation.

- **PiT+PiS Algorithm for MMCs with Device-Level IGBT Models**

  To extend the PiT algorithm to DC power grids and realize the device-level IGBT transient simulation under PiT, a PiT method that utilizes an ideal switch as the coarse-grid predictor and curve-fitting IGBT model as a fine-grid corrector is proposed and implemented on a multi-core CPU. A new TLM propagation-delay-based method is integrated into the PiT systems to enable flexibility regarding forming a PiT+PiS MTDC grid. The method is similar to the waveform relaxation and allows PiT subsystems to connect to traditional PiS subsystems. The case studies on a single MMC and CIGRÉ B4 system show a significant speed-up of the proposed methods and indicate great potential for hybrid PiT+PiS methods for device-level MTDC EMT simulation.

- **Heterogenous PiT+PiS Algorithm for TS-EMT Simulation**

  A heterogeneous CPU-GPU method that employs PiT+PiS algorithms for TS-EMT co-simulation is proposed. This novel approach integrates the Parareal algorithm on GPU frameworks alongside the conventional PiS algorithm, aiming for optimized parallel computation. The CPU schedules PiT operations while simultaneously executing GPU kernel functions. With advanced parallel computing technologies and asyn-

chronous memory management, this architecture not only enhances the scalability and extensibility compared to pure GPU models but also facilitates the simultaneous execution of EMT simulations on the CPU and transient stability simulations on the GPU. The case study demonstrates the method's proficiency, showcasing commendable accuracy and computational speed. The hybrid PiT+PiS methodology achieves approximately double the speed of the PiS-only method and astonishingly, a 165.6x acceleration compared to traditional sequential CPU computations for extensive systems. Moreover, this technique is easy to extend through multiple GPUs. The proposed hybrid heterogeneous model exhibits remarkable promises for addressing large-scale AC/DC power system simulation with PiT+PiS methods.

- **Data-Oriented Architecture for Cyber-Physical Smart Grid Simulation**

  Based on a data-oriented design philosophy, a novel data-oriented cyber-physical simulation platform for microgrids under the ECS framework is proposed to model the IEDs in a power system with flexible data components and extensible plugin architecture. Furthermore, a modern JSON-like MessagePack-based protocol is proposed for the vIEDs and is capable of completing various tasks needed for cyber-physical transient simulation. The results from the scenarios in the microgrid cluster study case show the accurate system behaviors and real-time performance of ECS-Grid. The IED systems and components can be extended to cyber-physical power dynamic or steady-state simulations thanks to the data-oriented design. The data-oriented ECS-Grid can inspire the renovation of industrial software tools and boost the digital twin research of the future CPPS. To further explore and demonstrate the effectiveness of a new data-oriented paradigm, an ECS-based data-oriented solution composed of RustSat, SatSDN, and ECSGrid is proposed to realize a real-time cyber-physical digital twin for a wide-area AC-DC grid based on LEO

satellite constellation network. The seamless integration between virtual IEDs in ECS-Grid and MiniNet's virtual network gateways for satellite networks has shown the full power of the new data-oriented architecture. The ECS-based simulation architecture has been investigated for smart grid simulation and has shown great potential for developing interdisciplinary and heterogeneous PiT+PiS computing applications.

- **AI-Enhanced Massively Parallel RES Models**

  The AI machine-learning-enhanced ANN models are developed to explore new parallel computing methods for nonlinear RES components in smart grids. Moreover, to realize flexible and fast massively parallel processing of these RES models in large-scale AC/DC power grid simulation, the integration between ANN models and traditional EMT simulation has been explored, which again proves the effectiveness of data-oriented ECS simulation architecture. The ANN models are accelerated by heterogeneous CPU-GPU hardware and GPU batching solutions, which achieve optimal performance as well as high flexibility. The proposed method has shown promising results for large-scale simulation, achieving high computational accuracy, decent GPU performance, and scalability across various system sizes. The machine-learning-based approaches may bring new approaches to increase the efficiency of PiS and PiT acceleration methods, especially for complex nonlinear components.

Through these research efforts, the boundaries of traditional power system transient simulation have been greatly expanded. Many previously unimaginable concepts have been transformed into reality through proposed innovative engineering methods. Through the proposed interdisciplinary and massively parallel PiT + PiS methodologies, the bright data-oriented future of smart grid simulation applications has been enlightened for an era of massively parallel data processing and AI technologies.

1.3 Thesis Outline

This thesis consists of eight chapters. The chapters are outlined as follows:

- **Chapter 2**
  introduces the theoretical foundations of AC/DC power grid EMT simulation and the PiT Parareal algorithm. Then, it presents the PiT algorithm implementation for AC Systems with TLMs and PiT+PiS Algorithm for MMCs with device-level IGBT models.

- **Chapter 3**
  introduces the theoretical foundations of transient stability simulation and GPU computing. Then it presents the heterogenous PiT+PiS algorithm for TS-EMT Simulation.

- **Chapter 4**
  introduces the ECS-Grid: data-oriented real-time simulation platform for cyber-physical power systems. Address the fundamentals to simulate a real-world IED in ECS architecture with physical EMT simulation.

- **Chapter 5**
  introduces the data-oriented implementation for cyber-physical power system simulation with LEO constellation networks.

- **Chapter 6**
  introduces the machine-learning-enhanced RES models and data-oriented massively parallel implementation on heterogeneous CPU-GPU hardware.

- **Chapter 7** provides the conclusions and suggestions for future works. The previous research works are concluded to highlight the research contributions of this PhD thesis. It also provides a future outlook on data-oriented PiT+PiS algorithms and AI accelerated simulation.

Chapter 2

Parallel-in-Time Object-Oriented Electromagnetic Transient Simulation of AC-DC Power Systems

2.1 Introduction

[1] This chapter demonstrate the initial efforts to utilize a PiT algorithm in EMT simulations for AC and DC power systems. The exploration of this innovative algorithm not only paves the way for subsequent research in parallel simulation techniques but also raises new scientific demands. These include the development of advanced software engineering solutions, new hardware acceleration strategies, and a deeper understanding of transient phenomena and modeling challenges.

The chapter begins with the fundamental theories of EMT simulation and Parareal algorithm. The EMT circuit DAE problems are described with graph-theory and matrix language, promoting a vectorized representation that aligns with modern parallel computing practices. This approach not only enhances the readability and scalability of the simulation code but also aligns with the

[1] This chapter includes works published in:

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

T. Cheng, T. Duan and V. Dinavahi, "Parallel-in-time-and-space electromagnetic transient simulation of multi-terminal DC grids with device-level switch modelling," *IET Generation, Transmission & Distribution,* Vol. 16, pp. 149-162, 2022

shift towards high-performance, massively parallel computing environments.

Then, it addresses the challenges of implementing PiT algorithms for transient components, such as handling complex transient models, TLMs with DDEs, interfacing issues, and the integration of detailed switching models in power electronics. These challenges are critical as they influence the simulation's accuracy and efficiency.

Through detailed case studies, the effectiveness of these methods in improving simulation speed and accuracy is demonstrated, emphasizing the practical benefits of these advanced computational techniques. Furthermore, the chapter outlines the theoretical underpinnings and implementation methods for these algorithms, providing a solid foundation for future advancements in the field.

2.2 Fundamentals of Power System EMT Simulation

In EMT simulation, dynamic physical components such as capacitors and inductors are represented by differential equations. Small and simple circuits such as RC, LC, and RLC can be solved with a form of ordinary differential equations or state-space format. However, a power grid usually consists of a large amount of different physical components, which is more suitable to solve as DAEs with nodal analysis [34]. The nodal analysis is based on solving a circuit equation system with nodal voltages as primary unknown variables. Using Kirchhoff's Current Law (KCL) any RLC circuits with k nodes can be represented by:

$$\begin{aligned}\sum \boldsymbol{i}_{out} &= \boldsymbol{i}_L + \boldsymbol{i}_C + \boldsymbol{i}_G \\ &= W_L \int \boldsymbol{v} dt + W_C \frac{d\boldsymbol{v}}{dt} + W_G \boldsymbol{v} = \boldsymbol{s}, \\ W_L = B_L^T [\frac{1}{\boldsymbol{L}}] B_L, W_C &= B_C^T [\boldsymbol{C}] B_C, W_G = B_G^T [\boldsymbol{G}] B_G,\end{aligned} \tag{2.1}$$

where B is the oriented incidence matrix whose rows correspond to the physical components and columns correspond to nodes, L is the inductance, C is the capacitance and G is the admittance, $\boldsymbol{s}$ is the vector of current injections by

sources. B is a transformation to gather the port voltages from global nodal voltages $\boldsymbol{v}$, while B^T can scatter the branch currents into the nodal injection vector. The W matrices are the weighted Laplacian matrices of different types of components, which are also called admittance matrices and play important roles in solving power grid equation systems. $[X]$ means diagonalized matrix of 1-D vector X.

To solve (2.1) with the Trapezoidal Rule, the following equations can be obtained:

$$\sum \boldsymbol{i}_{out_{n+1}} = \boldsymbol{i}_{L_{n+1}} + \boldsymbol{i}_{C_{n+1}} + \boldsymbol{i}_{G_{n+1}} = \boldsymbol{s}_{n+1}, \tag{2.2}$$

If the Trapezoidal Rule is applied to Equation (2.2), the nodal currents can be expressed by the following equations :

$$\begin{aligned} \boldsymbol{i}_{L_{n+1}} &= W_L \int_t^{t+\Delta t} \boldsymbol{v} dt + \boldsymbol{i}_L(t) \\ &\approx \frac{\Delta t}{2} W_L(\boldsymbol{v}_n + \boldsymbol{v}_{n+1}) + \boldsymbol{i}_{L_n} \\ \int_t^{t+\Delta t} \boldsymbol{i}_C dt &\approx \frac{\Delta t}{2}(\boldsymbol{i}_{C_{n+1}} + \boldsymbol{i}_{C_n}) = W_C(\boldsymbol{v_{n+1}} - \boldsymbol{v_n}) \\ \Rightarrow \boldsymbol{i}_{C_{n+1}} &\approx \frac{2}{\Delta t} W_C(\boldsymbol{v}_{n+1} - \boldsymbol{v}_n) - \boldsymbol{i}_{C_n}, \end{aligned} \tag{2.3}$$

where all variables at nth step are already known, which forms up following implicit DAEs:

$$\begin{aligned} (W_G + \frac{2}{\Delta t} W_C + \frac{\Delta t}{2} W_L)\boldsymbol{v}_{n+1} \quad &= \boldsymbol{s}_{n+1} + \boldsymbol{I}_{Leq}^{n+1} + \boldsymbol{I}_{Ceq}^{n+1}, \\ \boldsymbol{I}_{Leq}^{n+1} &= -\frac{\Delta t}{2} W_L \boldsymbol{v}_n - \boldsymbol{i}_{L_n}, \\ \boldsymbol{I}_{Ceq}^{n+1} &= \frac{2}{\Delta t} W_C \boldsymbol{v}_n + \boldsymbol{i}_{C_n}, \end{aligned} \tag{2.4}$$

where $\boldsymbol{I}_{Leq}^{n+1}$ and $\boldsymbol{I}_{Ceq}^{n+1}$ are synthetic current sources created by discretization. Notice that all $\boldsymbol{v}$ and $\boldsymbol{i}$ are corresponding to nodes instead of branches.

A similar rule is applied to all other dynamic or time-varying physical components in the power system for EMT simulation, which gives the general form:

$$\begin{aligned} Y\boldsymbol{v}_{n+1} &= \boldsymbol{s}_{n+1} + \boldsymbol{I}_{eq}^{n+1}, \\ Y &= \sum W_{diff}, \\ \boldsymbol{I}_{eq}^{n+1} &= \sum B^T g(B\boldsymbol{v}_n, B\boldsymbol{i}_n), \end{aligned} \tag{2.5}$$

where Y indicates the final admittance matrix for the system solution, which can be inverted to solve the primary unknown variable v_{n+1}; I_{eq}^{n+1} indicates all equivalent current sources generated by physical components; g is the discretized function to compute the $\boldsymbol{I}_{eq}^{n+1}$ of each component type; W_{diff} is the admittance matrix derived from differential equations such as $\frac{\Delta t}{2}W_L$ for inductors.

For nonlinear components such as diodes that have nonlinear voltage-current characteristics. (2.5) is extended to:

$$\begin{aligned} &\boldsymbol{i}_N + Y\boldsymbol{v}_{n+1} = \boldsymbol{s}_{n+1} + \boldsymbol{I}_{eq}^{n+1}, \\ &\boldsymbol{i}_N = B_N^T[f(B_N\boldsymbol{v}_{n+1})] \end{aligned} \tag{2.6}$$

where f is an element-wise nonlinear function of $B_N\boldsymbol{v}_{n+1}$. Newton's method is utilized to linearize the system and solve it, which converts Equation (2.6) into the following:

$$\begin{aligned} &F(\boldsymbol{v}_{n+1}) = \boldsymbol{i}_N + Y\boldsymbol{v}_{n+1} - (\boldsymbol{s}_{n+1} + \boldsymbol{I}_{eq}^{n+1}) = 0, \\ &F'(\boldsymbol{v}_{n+1}^m) = \frac{\delta \boldsymbol{i}_N}{\delta \boldsymbol{v}}|_{\boldsymbol{v}_{n+1}^m} + Y = J^m, \\ &\boldsymbol{v}_{n+1}^{m+1} = \boldsymbol{v}_{n+1}^m - \frac{F(\boldsymbol{v}_{n+1}^m)}{F'(\boldsymbol{v}_{n+1}^m)} = \boldsymbol{v}_{n+1}^m - J^{m^{-1}}F(\boldsymbol{v}_{n+1}^m), \end{aligned} \tag{2.7}$$

where m denotes the iteration index of the Newton method, J^m is the system Jacobian matrix at mth iteration. Equation (2.7) can be reorganized into:

$$\begin{aligned} J^m\boldsymbol{v}_{n+1}^{m+1} &= (J^m - Y)\boldsymbol{v}_{n+1}^m - \boldsymbol{i}_N^m + (\boldsymbol{s}_{n+1} + \boldsymbol{I}_{eq}^{n+1}) \\ &= W_{nl}^m\boldsymbol{v}_{n+1}^m - \boldsymbol{i}_N^m + (\boldsymbol{s}_{n+1} + \boldsymbol{I}_{eq}^{n+1}), \end{aligned} \tag{2.8}$$

where

$$W_{nl}^m = \frac{\delta \boldsymbol{i}_N}{\delta \boldsymbol{v}}|_{\boldsymbol{v}_{n+1}^m} = B_N^T[\nabla f(B_N\boldsymbol{v}_{n+1}^m)]B_N. \tag{2.9}$$

Therefore, the nonlinear components have the harmonized format of admittance matrices and artificial current injections, which gives the following recursion formula:

$$\begin{aligned} &J^m\boldsymbol{v}_{n+1}^{m+1} = \boldsymbol{s}_{n+1} + \sum \boldsymbol{I}_{eq}^{n+1} + \sum \boldsymbol{I}_{nleq}^{m+1}, \\ &\boldsymbol{I}_{nleq}^{m+1} = W_{nl}^m\boldsymbol{v}_{n+1}^m - \boldsymbol{i}_N^m. \end{aligned} \tag{2.10}$$

The Jacobian matrix needs to be assembled and inverted serval times in each simulation time step. Therefore, the f of many nonlinear components

may be converted into piece-wise linear function or use $\boldsymbol{v}_n$ to approximate $\boldsymbol{v}_{n+1}$ to speed up the computation. (2.1-2.10) cover the fundamentals of the EMT power system simulation.

2.2.1 Parareal Algorithm

The Parareal algorithm decomposes a large time interval into many small sub-intervals and solves the corresponding differential equations in parallel; therefore, it is considered to be an iterative multi-shooting approach [68]. As the differential equations present an initial value problem (IVP), initialization is mandatory for the solution process, and a fast serial predictor is used to provide the initial conditions, which divides the problem into a serial coarse-grid and a parallel fine-grid.

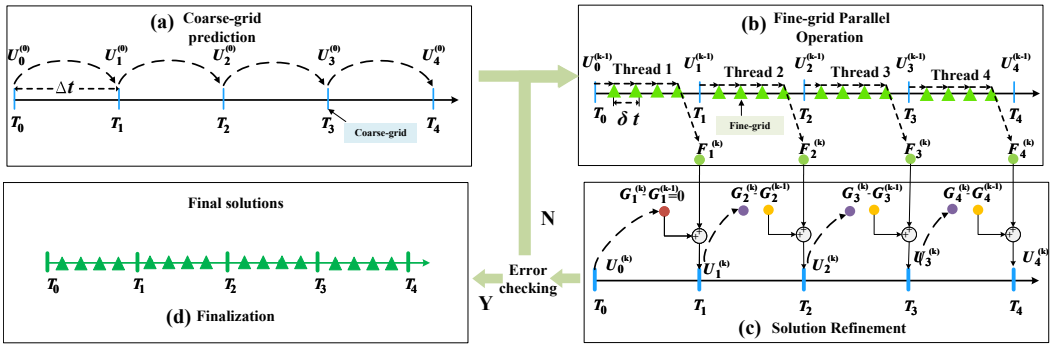

Figure 2.1: Sequence of operations in the Parareal algorithm: (a) Initialize $\boldsymbol{U}_j^{(0)}$ which equals to $\boldsymbol{G}_j^{(0)}$; (b) Produce fine-grid solution $\boldsymbol{F}_j^k$; (c) Refine $\boldsymbol{U}_j^{(k)}$ with $\boldsymbol{U}_j^{(k)} = \boldsymbol{G}_j^{(k)} + \boldsymbol{F}_j^k - \boldsymbol{G}_j^{(k-1)}$, then continue (b) to start a new iteration.

The Parareal algorithm for overall N intervals can be expressed by the following system of nonlinear equations:

$$\boldsymbol{E}(\boldsymbol{U}) := \begin{cases} \boldsymbol{U}_1 - \boldsymbol{F}\big(T_1, T_0, \boldsymbol{U}_0\big) = 0, \\ \boldsymbol{U}_2 - \boldsymbol{F}\big(T_2, T_1, \boldsymbol{U}_1\big) = 0, \\ \qquad \vdots \\ \boldsymbol{U}_N - \boldsymbol{F}(T_N, T_{N-1}, \boldsymbol{U}_{N-1}) = 0, \end{cases} \tag{2.11}$$

where $\boldsymbol{U}_0$ is a vector containing the known initial values, $\boldsymbol{U}_j (j = \{1, 2..., N\})$ are the final solution produced by a fine solution operator $\boldsymbol{F}(T_j, T_{j-1}, \boldsymbol{U}_{j-1})$;

T_j is the time instant of the N sub-intervals. By applying Newton's method,

$$(\boldsymbol{U}^{(k)} - \boldsymbol{U}^{(k-1)})\frac{d}{d\boldsymbol{U}}\boldsymbol{E}(\boldsymbol{U}^{(k-1)}) = \boldsymbol{E}(\boldsymbol{U}^{(k-1)}), \tag{2.12}$$

the following iterative equation for each $\boldsymbol{U}_j$ is obtained:

$$\begin{aligned}\boldsymbol{U}_j^{(k)} = &\boldsymbol{F}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right) \\ &+ \frac{\partial \boldsymbol{F}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right)}{\partial \boldsymbol{U}_{j-1}^{(k-1)}}\left(\boldsymbol{U}_{j-1}^{(k)} - \boldsymbol{U}_{j-1}^{(k-1)}\right),\end{aligned} \tag{2.13}$$

where $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$ is the derivative of operator function $\boldsymbol{F}$, written in a discrete form as:

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}} = \frac{\boldsymbol{F}^{(k)} - \boldsymbol{F}^{(k-1)}}{\boldsymbol{U}^{(k)} - \boldsymbol{U}^{(k-1)}}. \tag{2.14}$$

If $\boldsymbol{F}$ is used to obtain $\boldsymbol{F}^{(k)}$, it is identical to the normal sequential solution with $\boldsymbol{F}$. However, the Quasi-Newton method can be used to approximate the Jacobian, which can be naturally done by an operator $\boldsymbol{G}$ with a larger time-step, so that the derivative can be generated to make the PiT computing feasible:

$$\begin{aligned}\boldsymbol{F}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k)}\right) &\approx \boldsymbol{G}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k)}\right) \\ \boldsymbol{F}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right) &\approx \boldsymbol{G}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right).\end{aligned} \tag{2.15}$$

The discrete sparse time points processed by $\boldsymbol{G}$ at $\{T_1, T_2, ..., T_N\}$ constitute the coarse-grid; The discrete time points processed by $\boldsymbol{F}$ in $[T_1, T_N]$ constitute the fine-grid. Since the serial $\boldsymbol{G}$ produced the approximations for each $[T_{j-1}, T_j]$ sub-interval, $\boldsymbol{F}$ can execute them in parallel.

$\boldsymbol{G}$ can also be a faster integration method like Euler and backward Euler method while $\boldsymbol{F}$ is a more time-consuming method such as Trapezoidal and Runge-Kutta methods. In this work, the Trapezoidal integration method is used, which gives

$$\begin{aligned}\boldsymbol{F}^k &= \boldsymbol{x}_n^k + \frac{1}{2}h_F(\boldsymbol{f}(\boldsymbol{x}_{n+1}^k, \boldsymbol{u}_{n+1}^k) + \boldsymbol{f}(\boldsymbol{x}_n^k, \boldsymbol{u}_n^k)), \\ \boldsymbol{G}^k &= \boldsymbol{x}_n^k + \frac{1}{2}h_G(\boldsymbol{f}(\boldsymbol{x}_{n+1}^k, \boldsymbol{u}_{n+1}^k) + \boldsymbol{f}(\boldsymbol{x}_n^k, \boldsymbol{u}_n^k)), \quad h_F < h_G,\end{aligned} \tag{2.16}$$

where $\boldsymbol{f}$ refers to aforementioned equations for EMT components, h_F and h_G are the fine-grid time-step and coarse-grid time-step, respectively. The only difference between $\boldsymbol{F}$ and $\boldsymbol{G}$ is the size of time steps and they are the

same Trapezoidal Rule. Substituting $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$ with the approximation, the following equation is obtained to conduct the predict-correct iteration between coarse and fine grids:

$$\begin{aligned}\boldsymbol{U}_j^{(k)} = \boldsymbol{F}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right) \\ + \boldsymbol{G}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k)}\right) - \boldsymbol{G}\left(T_j, T_{j-1}, \boldsymbol{U}_{j-1}^{(k-1)}\right),\end{aligned} \tag{2.17}$$

which was proven to be a Quasi-Newton method in [69].

As shown in Fig. 2.1, the Parareal algorithm has four major progressions: (a) *Initialization*: the initial guess is generated from the coarse operator; (b) *Fine-grid Parallel Operation*: the fine-grid workers produce the fine-grid solutions concurrently from the initial values of coarse-grid; (c) *Solution Refinement*: the coarse-grid operator produces new predictions and correct the solutions at T_j using (2.17); (d) *Finalization*: If the error is smaller than tolerance, the fine-grid workers generate the final converged solutions. Otherwise, it continues to step (b) and continues the Parareal iterations.

2.2.2 Fundamental AC Power System Models

The equations are formed from different components in the power system. Most components in EMT simulation are treated as an equivalent conductance in parallel with a current source, which is convenient for nodal analysis. However, the behavior of these components differs very much from each other. It is more convenient to integrate their states into their local space, and the only common states they share are the node voltages.

Transformer Model The single-phase transformers in EMT simulation are composed of an equivalent circuit and an ideal transformer. Due to the structure of transformers, the model is more complicated than simple RLC components, but the final derived EMT models have the same form of admittance matrices and equivalent current sources. The transformer model in Fig. 2.2 can be expressed as

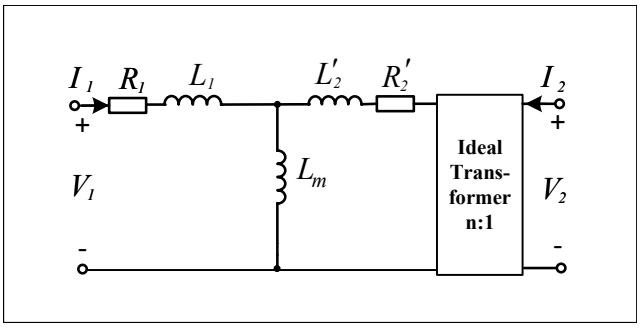

Figure 2.2: Equivalent circuit model of single-phase transformers for EMT simulation.

$$
\begin{aligned}
&T\boldsymbol{V} = RT^{-1}\boldsymbol{I} + LT^{-1}\frac{d}{dt}\boldsymbol{I} \\
&\boldsymbol{V} = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}, \quad \boldsymbol{I} = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix}, R = \begin{bmatrix} R_1 & 0 \\ 0 & R_2' \end{bmatrix}, \\
&L = \begin{bmatrix} L_m + L_1 & L_m \\ L_m & L_m + L_2' \end{bmatrix}, \quad T = \begin{bmatrix} 1 & 0 \\ 0 & n \end{bmatrix},
\end{aligned}
\tag{2.18}
$$

where V_1 and I_1 represent the port voltage and current for the primary winding, respectively; V_2 and I_2 represent the port voltage and current for the secondary winding; R and L denote the well-known resistance and inductance of a transformer, which can be determined from standard parameters provided by manufacturers; R_2' and L_2' represent the resistance and inductance converted to the primary winding; T is the transformation matrix to convert secondary-side voltages to the primary side, and n is the transformer turns ratio. It is noteworthy that $\boldsymbol{V}$ and $\boldsymbol{I}$ are not transformed, while the admittances are computed at the primary winding side.

With Trapezoidal Rule, (2.18) can be discretized to the following

$$
\begin{aligned}
&(\frac{2}{\Delta t}L_T + R_T)\boldsymbol{I}_{n+1} = \boldsymbol{V}_{n+1} + \boldsymbol{V}_n + (\frac{2}{\Delta t}L_T - R_T)\boldsymbol{I}_n \\
&R_T = T^{-1}RT^{-1}, L_T = T^{-1}LT^{-1},
\end{aligned}
\tag{2.19}
$$

and it can be organized into the following

$$
\begin{aligned}
&\boldsymbol{I}_{n+1} = G_T \boldsymbol{V}_{n+1} + \boldsymbol{I}_{eq_{n+1}}, \\
&\boldsymbol{I}_{eq_{n+1}} = G_T \boldsymbol{V}_n + F_T \boldsymbol{I}_n, \\
&G_T = (\frac{2}{\Delta t} L_T + R_T)^{-1} = T(\frac{2}{\Delta t} L + R)^{-1} T, \\
&F_T = G_T(\frac{2}{\Delta t} L_T - R_T) = TG(\frac{2}{\Delta t} L - R) T^{-1},
\end{aligned}
\tag{2.20}
$$

which derives the EMT model for single-phase transformers. The final step is to convert the admittances and currents to the coordinates of nodal analysis since the transformer uses port voltages and mesh currents instead of branch currents and nodal voltages.

With this single-phase model, three-phase transformers are built by connecting single-phase transformers according to the desired connection groups.

Generator Model The synchronous generators are represented by the machine model with one kd winding and two kq windings. The relationship between voltages and currents can be expressed as:

$$
\boldsymbol{v}_{um}(t) = \boldsymbol{R}_{um} \boldsymbol{i}_{um}(t) - \frac{d}{dt} \boldsymbol{\psi}_{um}(t) + \boldsymbol{u}(t) \tag{2.21}
$$

$$
\boldsymbol{\psi}_{um}(t) = \boldsymbol{L}_{um} \boldsymbol{i}_{um}(t), \tag{2.22}
$$

where $\boldsymbol{v}_{um} = [v_d, v_q, v_0, v_f, 0, 0, 0]^{\mathrm{T}}$, $\boldsymbol{i}_{um} = [i_d, i_q \ i_0, i_f, i_{kd}, i_{kq1}, i_{kq2}]^{\mathrm{T}}$, $\boldsymbol{\psi}_{um} = [\psi_d, \psi_q, \psi_0, \psi_f, \psi_{kd} \ \psi_{kq1}, \psi_{kq2}]^{\mathrm{T}}$, $\boldsymbol{u} = [-\omega\psi_q, \omega\psi_d, 0, 0, 0, 0, 0]^{\mathrm{T}}$, $\mathbf{R}_{um} = \mathrm{diag}\,(R_d, R_q, R_0, R_f, R_{kd}, R_{kq1}, R_{kq2})$ and $\mathbf{L}_{um}$ is the leakage inductance matrix.

Indirect approaches are widely used in the EMT program to interface the synchronous machines. For example, the Norton current source representation of machines implemented in PSCAD/EMTDC®. However, this kind of model has weak numerical stability [70], and causes oscillations in Parareal iterations. Therefore, a machine model from [71], which can be used in variable time-stepping methods, is used in the proposed parallel-in-time simulation.

The machine is represented by a Thevenin voltage source. discretized for-

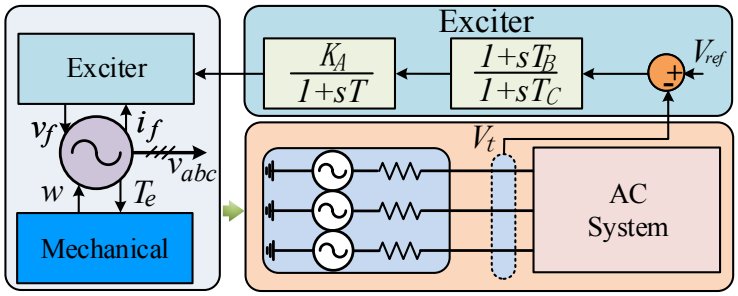

Figure 2.3: Synchronous generator representation using variable time-stepping machine model.

mulation of (2.21) under Trapezoidal Rule is given by

$$\boldsymbol{v}_{um}^{n+1} = -[\boldsymbol{R}_{um} + \boldsymbol{L}_{eq} - \omega \boldsymbol{L}_u]\boldsymbol{i}_{um}^{n+1} + \boldsymbol{v}_{hist} \tag{2.23}$$

$$\boldsymbol{v}_{hist} = \boldsymbol{u}^n - \boldsymbol{v}_{um}^n - [\boldsymbol{R}_{um} - \boldsymbol{L}_{eq}]\boldsymbol{i}_{um}^n \tag{2.24}$$

where $\boldsymbol{L}_{eq} = \frac{2}{\Delta t}\boldsymbol{L}_{um}$, $\boldsymbol{u}^n = \omega \boldsymbol{L}_u \boldsymbol{i}_{um}^n$, and $\boldsymbol{L}_u = [\boldsymbol{L}_{um}(2); \boldsymbol{L}_{um}(1); \boldsymbol{0}; \boldsymbol{0}; \boldsymbol{0}; \boldsymbol{0}; \boldsymbol{0}]$.

The derivation process is detailed in [71]. Since the mechanical state ω changes slower than electrical ones, the numerical stability is better than the models that make relaxations or assumptions on electrical state variables. In the Parareal iteration, the state vector for this model is $\{\boldsymbol{v}_{um}, \boldsymbol{i}_{um}, \boldsymbol{\psi}_{um}, \omega\}$.

Transmission Line Model The transmission lines are the most common components in a power system. However, the models are based on the traveling wave theory which means that the solutions are dependent on a range of past states. This brings DDEs to the EMT [72].

Taking the lossless line as an example, the equations to update historical current source are given as:

$$\begin{aligned} i_m^{hist}(t) &= -2Gv_k(t-\tau) - i_k^{hist}(t-\tau) \\ i_k^{hist}(t) &= -2Gv_m(t-\tau) - i_m^{hist}(t-\tau). \end{aligned} \tag{2.25}$$

where τ is the transmission delay i_m^{hist} is the receiving-end current source, i_k^{hist} is the sending-end current source and G is the characteristic conductance of the transmission line. Details can be found in [34]. Since the equations are in continuous-time domain, to work with discrete-time integration, linear

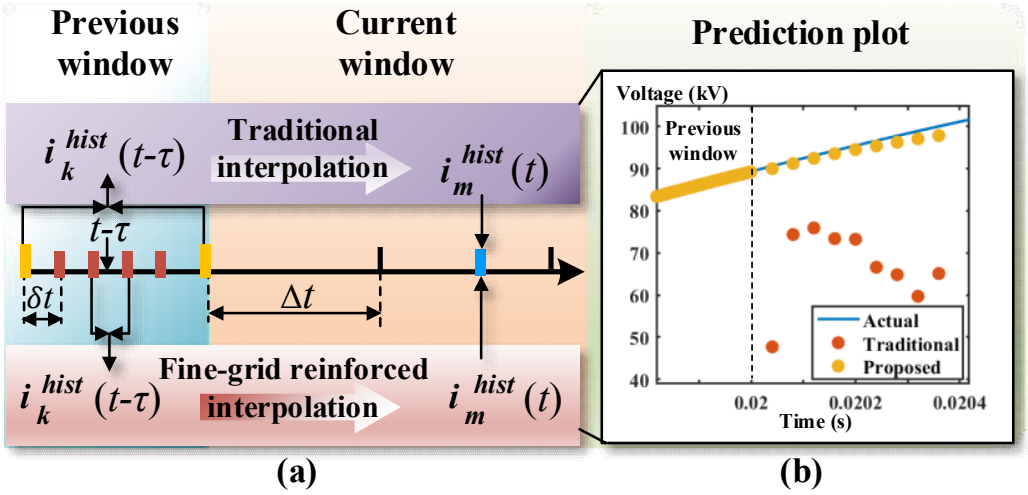

Figure 2.4: (a) Proposed interpolation scheme; (b) Prediction results comparison.

interpolation is used to get the approximation between two discrete time points near $t - \tau$. However, the traditional method cannot work under the parallel-in-time scenario for two reasons.

First, the DDE problems are not considered in previous parallel-in-time research [72], [73]. Although Parareal algorithm can solve nonlinear DAE problems by predicting states at certain time points in coarse-grid and refine them in fine-grid, it is difficult to do such thing for a transmission line because the historical states for the fine-grid transmission lines do not exist in coarse-grid. Using interpolation to predict the fine-grid history vectors cannot reflect the transient waveform of the discrete system with a smaller time-step. In this case, all transmission line history data should be prepared before Parareal iteration. Currently, limiting the time window of iteration is the only way to avoid this dependency issue.

Second, limiting the time window is still not enough. The traditional transmission line uses linear interpolation to approximate the historical value at $t - \tau$. However, the approximated historical values are inconsistent with the fine-grid ones. As shown in Fig. 2.4 (b), the traditional interpolation's inconsistency causes a huge error between coarse and fine-grid so that the prediction in the next window fails to meet the assumption in (2.15) and causes deviations.

To solve these problems, a fine-grid reinforced transmission line model implementation is proposed, which is shown in Fig. 2.4 (a). Unlike conventional line models, where each line has its history vector to cover only one delay cycle, a transmission line in fine-grid and coarse-grid share the same memory for historical data. The computation is set to a time window which is smaller than the transmission delay τ, so that accurate historical data are prepared from the previously completed window, which avoids the data dependency issue. To improve the coarse-grid prediction, the coarse-grid transmission lines must read data from the data points in fine-grid history vectors, which requires a conversion between two time-steps to get the data index, given by

$$j_{fine} = floor(j_{coarse}\frac{\Delta t}{\delta t} + (\frac{\Delta t}{\delta t} - 1)\frac{\tau}{\Delta t}), \tag{2.26}$$

where j_{fine} is the converted index in fine-grid history vector for coarse-grid transmission line; j_{coarse} is the original index in Δt step simulation; δt is the fine-grid time-step, and the index is always truncated to an integer.

2.3 Parareal Application To Power System EMT Simulation

Unlike other research works on parallel-in-time simulation, which focuses on specific differential equations, the power system EMT simulation needs to handle different kinds of equations and configurations. Also, Parareal algorithm requires the ability to restart the simulation at an arbitrary time to do iterations. Therefore, it is necessary to model the power system on a higher-level abstraction. A component-based system architecture is proposed to handle the parallel-in-time complexity flexibly and elegantly, and serves as the fundamental element to build the parallel-in-time simulation program.

Component-Based System Architecture A circuit is an undirected graph topological relationship between different components, where the components contain all the edge information of the graph. Therefore, the circuit can be represented with a vector of components. As shown in Fig. 2.5, the object-oriented concept is used to model the circuit system and components. Major

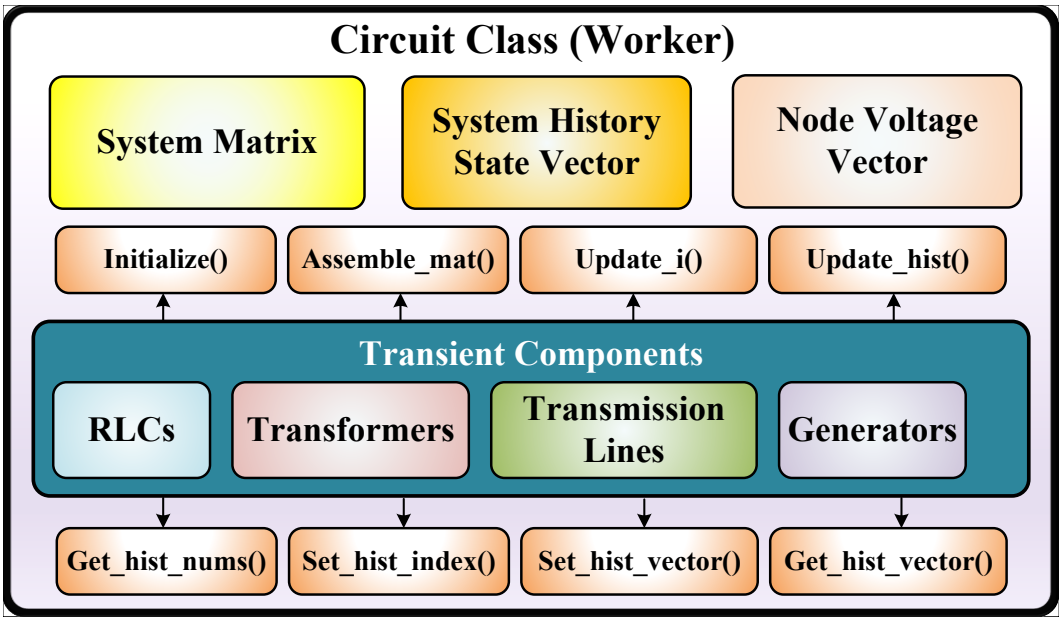

Figure 2.5: Circuit class architecture for the proposed parallel-in-time EMT simulation program.

properties of the *circuit* class contain the system matrix, state vectors, and a container for heterogeneous dynamic time-varying components.

All the components inherit from abstract base component class: *TransientComponent* so that the circuit object is able to call individual component functions with standard interface: *initialize(); assemble_mat(); update_i(); update_hist()*, with object polymorphism features. The *get_hist_nums(); set_hist_index(); set_hist_i(); get_hist_vector()* functions are interface to gather or scatter state variables between individual components to the system history state vector.

The algorithm flow chart is shown in Fig. 2.6. Before solving the circuit, each component initializes its variables and equivalent conductance according to the time-step. Then, they assemble the global system matrix according to their branch information in the graph. With the matrix formed, the circuit is ready to be solved. With this architecture, the complexity of power system EMT models is encapsulated into two functions of circuit class, the *init()* and *step()*. Therefore, the component-based object-oriented architecture has significantly simplified the algorithm implementation for a power system with

various types of transient components.

The basic version of Parareal algorithm is by using multiple circuit instances proposed in the previous section as workers. The algorithm has four stages concluded in Fig. 2.7. The windowed algorithm is the practical implementation because it won't allocate all memories at once.

Table 2.1: Performance Comparison of Different Test Cases with Fixed $T_{window} = 200\mu s$, $\Delta t = 40\mu s$ and $\delta t = 1\mu s$ for A $500ms$ Duration Using 5 Threads.

Method	IEEE-9	IEEE-39	IEEE-118
Parareal (s)	4.019	25.233	283.393
Parallel LU (s)	4.756	46.641	385.024
Sequential (s)	4.507	44.579	591.015
Parareal Speed-up	1.12	1.77	2.09
Theoretical Speed-up	1.88	2.18	2.24

2.4 Case Studies

A CPU-based parallel-in-time EMT program is implemented in C++ with Intel® Threading Building Block (Intel® TBB) and Intel® Math Kernel Library (Intel® MKL). Tests are performed on the IEEE-9, IEEE-39 and IEEE-118 test systems to verify the parallel-in-time results against the sequential ones. In addition, the parallel-in-time performance is compared to a traditional spatial parallel computing implementation which utilized Intel® MKL's highly optimized parallel lower-upper (LU) decomposition algorithm.

The parameters of test cases are from [74]. The same thread number and algorithm configuration are used to compare the performance under the same condition. The time window and other parameters are shown in Table 2.1. For the IEEE-9 system, the minimal delay of transmission lines is $246\mu s$, which is enough for the $200\mu s$ time window. But for the other test systems, there are some short transmission lines. The transmission line length in IEEE-39 is scaled-up to 60-100km so that it can fit the time window and get reason-

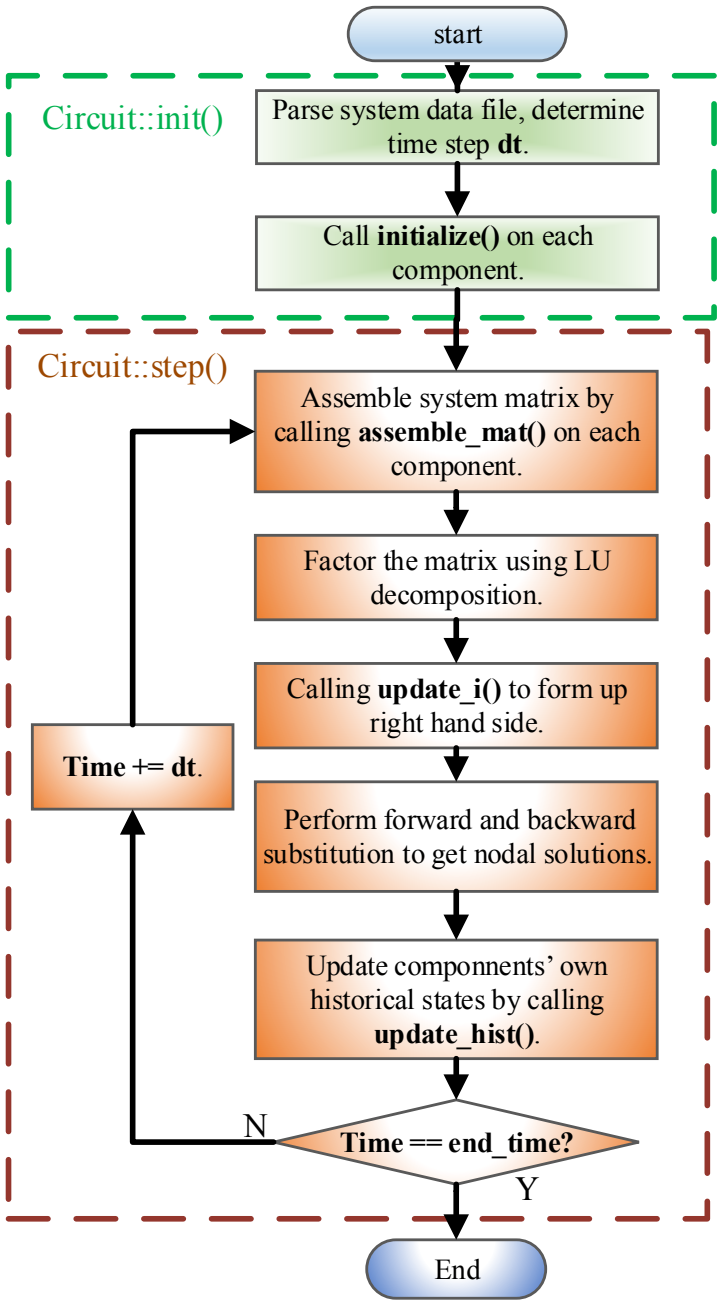

Figure 2.6: Flowchart of proposed parallel-in-time EMT simulation program.

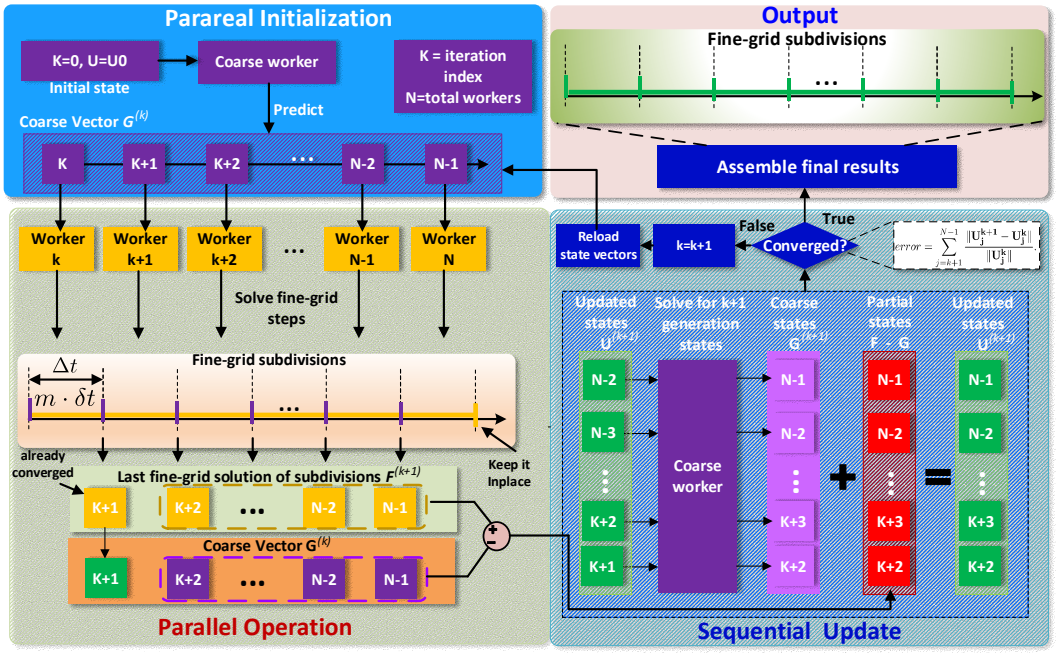

Figure 2.7: Detailed procedures in the proposed parallel-in-time EMT simulation.

able speed-up. For the IEEE-118 case, the transmission lines below 60km are simplified to multiple-PI sections and the remaining 61 lines are modeled with the Bergeron model. Without the simplification, although good results can be obtained, the parallel-in-time algorithm falls back to the sequential program.

A three-phase-to-ground fault happens at 0.3s, Bus 8 in IEEE-9, and 0.3s, Bus 14 in IEEE-39 case. A fault happens at 0.2s in the IEEE-118 case at Bus 38. The fault resistor size is $R_{fault} = 0.01\Omega$, $R_{clear} = 1\text{M}\Omega$. To achieve stable and correct results, the error tolerance of a whole time window is set to 0.01, which means the relative error sum of coarse steps cannot exceed 1%. Parallel workers are fixed to five for parallel-in-time cases, and they use the sequential LU algorithm. As shown in Fig. 2.8, the parallel-in-time simulation results coincide with those from the traditional method, and the zoomed-in views show the expected high accuracy. The results are verified with PSCAD/EMTDC®.

To evaluate the performance, the theoretical workload is defined by the simulation time consumption without any overhead from Parareal iterations or thread synchronizations. To achieve speed-up, the parallel-in-time workload must be smaller than the sequential one to get speed-up. The workload ratio

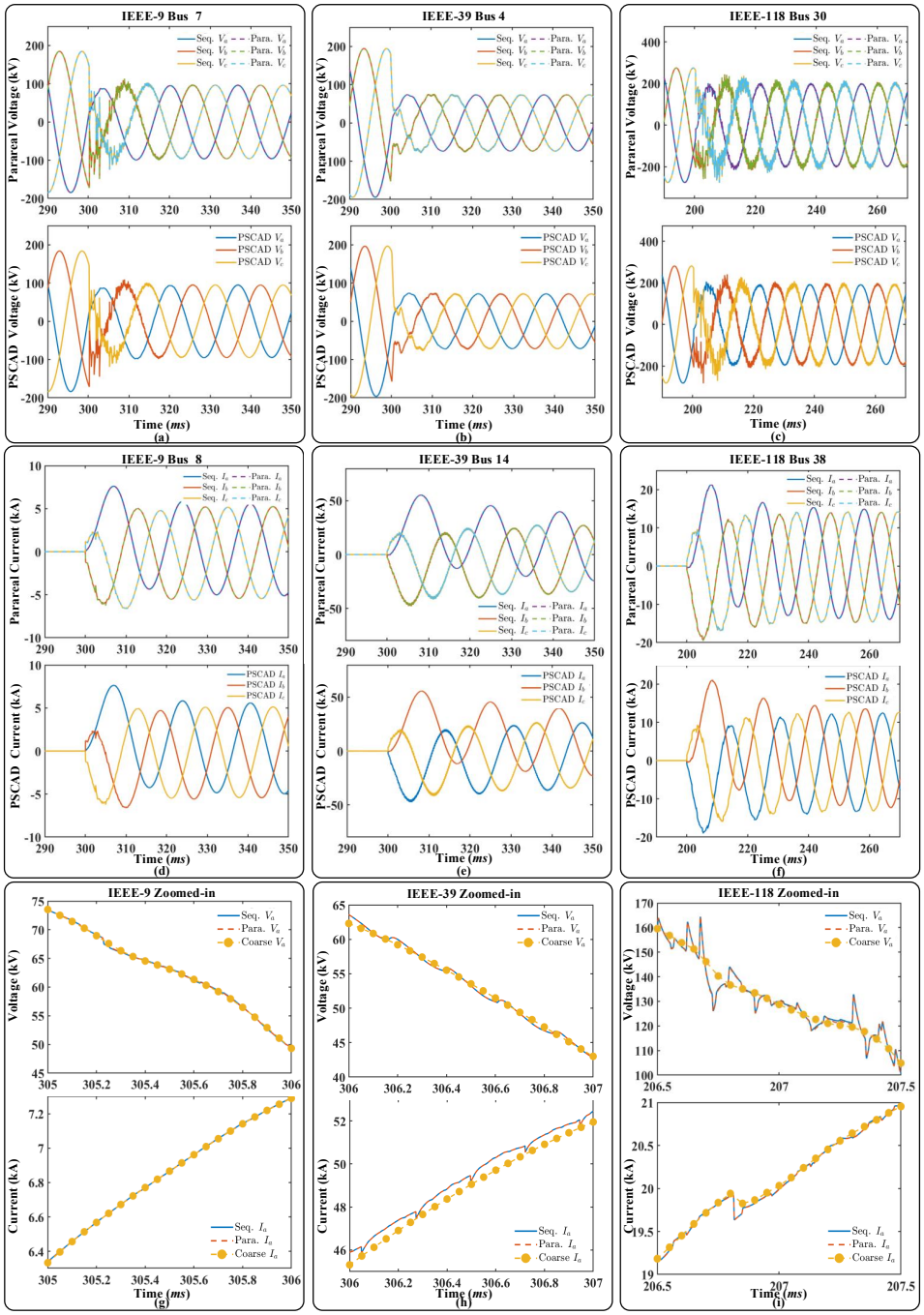

Figure 2.8: Simulation results of three-phase voltages: (a) Bus 7 in IEEE-9; (b) Bus 4 in IEEE-39; (c) Bus 30 in IEEE-118. Simulation results of fault currents: (d) Bus 8 in IEEE-9; (e) Bus 14 in IEEE-39; (f) Bus 38 in IEEE-118. Zoomed-in comparison: (g) voltages and currents of IEEE-9; (h) voltages and currents of IEEE-39; (i) voltages and currents of IEEE-118.

Table 2.2: Comparison of Sequential, Parallel LU, and parallel-in-time IEEE-118 Simulation with Various T_{window}, Δt and fixed $\delta t = 1\mu s$ for a $300ms$ Duration Using 5 Threads.

Configuration	Average iteration	Simulation time (s)	Speed-up	Theoretical speed-up	Eff.
$400\mu s, 80\mu s$	2.98	236.783	1.50	1.59	30%
$200\mu s, 40\mu s$	1.99	170.034	2.08	2.24	42%
$125\mu s, 25\mu s$	1.96	180.244	1.97	2.10	39%
$50\mu s, 10\mu s$	1.00	218.468	1.62	1.74	32%
Parallel LU	-	230.530	1.54	-	31%
Sequential	-	354.108	1.00	-	-

can be obtained by T_{par}/T_{seq} , where T_{par} is the parallel workload and T_{seq} is the sequential workload, and the theoretical speed-up is the reciprocal of workload ratio. Also, the parallel efficiency is computed to evaluate the utilization of parallel processors. The efficiency is the ratio of the speed-up to the number of threads.

Table 2.3: Performance Comparison of Various Thread Number with Fixed $\Delta t = 40\mu s$ and $\delta t = 1\mu s$ for A $500ms$ Duration.

Thread	Theoretical Speed-up			Simulation Time (s)			Actual Speed-up		
Number	IEEE-9	IEEE-39	IEEE-118	IEEE-9	IEEE-39	IEEE-118	IEEE-9	IEEE-39	IEEE-118
16	2.27	1.43	1.93	8.63	52.79	480.76	0.52	0.84	1.23
12	1.87	1.74	1.91	7.52	39.26	408.86	0.60	1.13	1.45
8	2.18	1.74	1.83	4.40	34.50	340.75	1.03	1.29	1.73
6	2.14	2.21	2.55	3.77	25.16	256.80	1.20	1.76	2.30
5	2.14	2.18	2.24	3.63	25.23	287.95	1.25	1.76	2.05
4	1.79	1.79	1.83	4.08	29.52	334.33	1.11	1.50	1.77

Normally a small scale system cannot benefit from parallel computing due to the thread launching and synchronization overhead. As shown in Table 2.1, the Parallel LU cannot achieve speed-up under the IEEE-9 (matrix size 27×27) and IEEE-39 (matrix size 117×117) case, while in IEEE-118 (matrix size 354×354) case the speed-up is obvious. In contrast, the Parareal algorithm can get speed-up in all three cases because the speed-up is mainly determined by the temporal factors, which is more obvious in Table 2.3.

Table 2.2 shows the performance of the IEEE-118 simulation with the same fine-grid time-step $\delta t = 1\mu s$ and different coarse-grid time-steps. The parallel-in-time results are compared to sequential, parallel LU, and theoretical speed-up. The sequential and parallel LU simulation uses the same time-step $\delta t = 1\mu s$. All parallel-in-time simulation's error tolerances are set to 0.01 per time window so their results are on the same accuracy level.

From Table 2.2, the best performance case is when the time window is 200μs. In this case, it is 2.08x faster than the sequential one while the parallel efficiency is 42%. The speed-ups are close to the theoretical ones and the overhead is around 10%, indicating that the implementation is highly efficient. The actual speed-ups and parallel efficiency are higher than the MKL parallel LU case except for the 400μs case.

Table 2.3 shows the performance of all test cases with different thread numbers. The Δt, and δt remain constant for all thread numbers and test cases, so the time window size changes with the thread number. The different test cases show similar theoretical speed-ups. If the coarse prediction can match the fine-grid results on a wider range, the speed-up can be higher. However, in power system EMT simulations, the delay of transmission lines restricts the time window size so utilizing more threads may not get better performance. The exception is the IEEE-9 case, which gets the best theoretical speed-up with 16 threads. In general, the speed-up of the Parareal method is mainly affected by the system's time-domain characteristics such as the model's time constants and simulation time-steps rather than the system's scale.

Although the time-domain characteristics are dominant, the overheads of multi-thread synchronization and error evaluations are noticed in the smallest test case. The actual speed-up is much far away from the theoretical ones in the IEEE-9 case compared to larger cases, indicating much room for improvement in small scale systems. When the thread number equals 16, the CPU cannot finish all the jobs in parallel so the speed-up goes down significantly for all cases. The optimal thread number for these cases is 5 or 6, which gives a time window of 200-240μs. This is exactly the minimal transmission line delay set

for all the test cases. This indicates that the transmission line delays are the main factors to affect the best speed-up we can get. exceeding this boundary causes inaccurate predictions so the speed-up drops down. Therefore, the treatment of the delays in transmission lines is significant for parallel-in-time power system EMT simulation. For IEEE-118 case, the best speed-up is 2.30x and the parallel efficiency is 38.3%, which is still higher than parallel LU decomposition.

2.5 MMC PiT Modeling

The PiT implementation of power electronic devices involves in various types of components including discrete switching events and the consistent behavior between coarse-grid and fine-grid control systems. The following sections will introduce how the switches, converters and control systems in the MMC model is adapted for PiT algorithms.

Three-Phase MMC Modeling As shown in Fig. 2.9 (a), the submodules of the MMC are solved as individual sub-circuits with the arm current of the previous time-step as the injection $i_{arm}(n-1)$ after applying the $\boldsymbol{V}$-$\boldsymbol{I}$ decoupling method [75], [76], which is valid since the time constant of the submodule is much larger than the simulation time-step – normally a few microseconds.

The fact that pure voltage sources cannot be used in nodal analysis directly prompts the merging of an arm inductor into the corresponding arm and results in the reduced equivalent model in Fig. 2.9 (b). The reduced equivalent current source and admittance are used to construct the single-phase or three-phase MMC topology shown in Fig. 2.9 (c), which finally yields a 5x5 element in nodal analysis. The nodal voltages of Node 1-5 in Fig. 2.9 (c) are the basic state variables that should participate in Parareal iteration.

The nearest-level-modulation (NLM) [77] is in charge of MMC internal current flow so as to maintain stable submodule capacitor voltages. Based on the proposed MMC architecture, the ideal switch and device-level transient

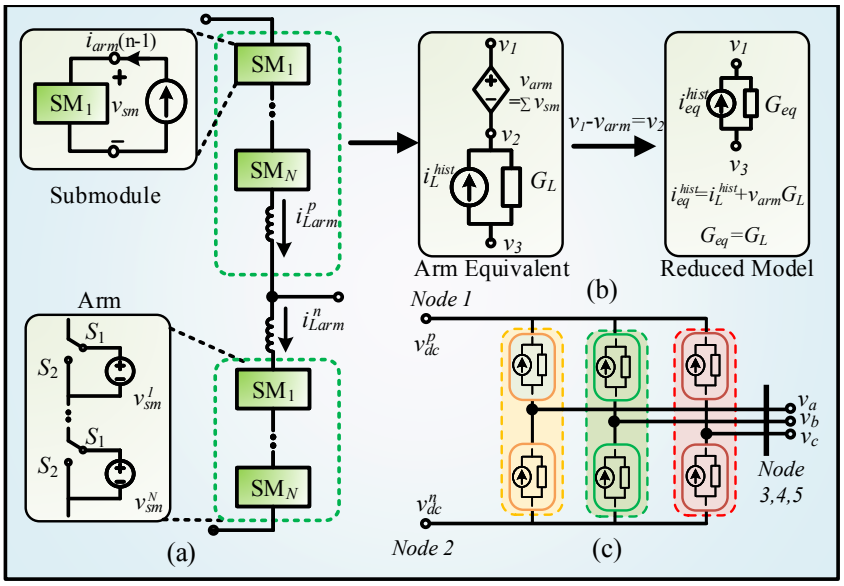

Figure 2.9: The three-phase MMC model for nodal analysis. (a):MMC arm simplification by $\boldsymbol{v}$-$\boldsymbol{i}$ coupling (b):Arm nodal reduction (c):Three-phase MMC equivalent circuit

curve-fitting half-bridge submodules (HBSMs) are investigated in this work.

Ideal Switch Model The IGBTs and diodes in the ideal switch model are represented by fixed on-state and off-state resistors so it is considered as an ideal switch model. The discrete numerical integration methods will cause overshoots at the switching instant due to the discrete nature, which cause numerical oscillation under the Trapezoidal Rule. Therefore, the blocking state requires additional measurements to handle the numerical oscillations. Some measurements are presented in [76], but here a common interpolation method used by PSCAD® is introduced [78]. As shown in Fig. 2.10, the interpolation method takes four steps to avoid the oscillation: (1) the system detects the zero-crossing event at $t = \Delta t$, then the system move backwards and all the state variables such as voltages and currents are interpolated to the zero-crossing time instant $t = t_0$, using the linear interpolation; t_0 should be the exact switching time between $t = 0$ and $t = \Delta t$; (2) using the system state approximated at $t = t_0$, the system triggers the diode switching and solve the solution at $t = t_0 + \Delta t$; (3) since the $t_0 + \Delta t$ is not at the original discrete-time grid, the system performs another linear interpolation with solutions at $t = t_0$

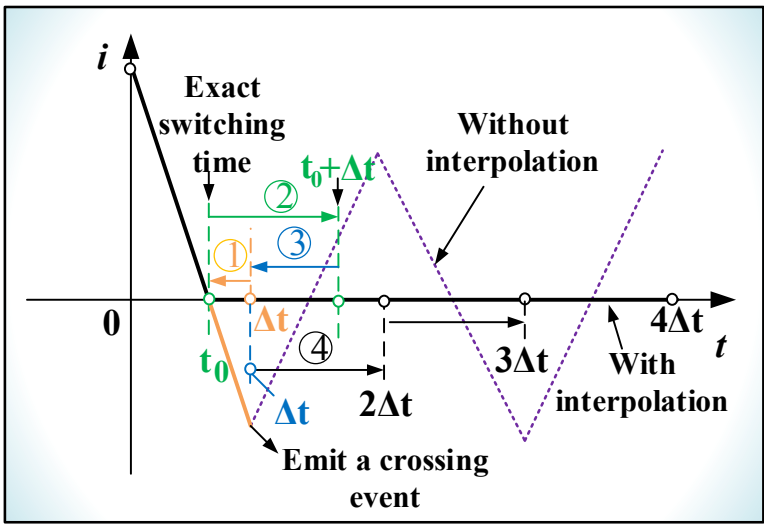

Figure 2.10: Interpolation approach to eliminate numerical oscillation caused by uncontrolled diode switching in the MMC model [78], [79].

and $t = t_0 + \Delta t$ to get the solution at $t = \Delta t$; (4) with the corrected solution at $t = \Delta t$, the system can go back to the normal solution steps.

As mentioned in Section 2.5, the individual submodule is solved independently from the main circuit, which means for ideal switch model it needs to solve a bunch of 2x2 matrices to get the capacitor voltages $\boldsymbol{v}_c$ and submodule port voltages $\boldsymbol{v}_{sm}$, which can be expressed by following discretized equations under nodal analysis:

$$\begin{bmatrix} g_{sw1} + g_{sw2} & -g_{sw1} \\ -g_{sw1} & g_{sw2} \end{bmatrix} \begin{bmatrix} v_c(n) \\ v_{sm}(n) \end{bmatrix} = \begin{bmatrix} i_{ceq}(n) \\ i_{arm}(n-1) \end{bmatrix}, \\ i_{ceq}(n) = v_c(n-1)G_{ceq} + i_c(n-1), \tag{2.27}$$

where G_{ceq}, i_{ceq} denotes the capacitor equivalent admittance and current source, $g_{sw1,2}$ denote the equivalent admittance of ideal switches, i_c is the capacitor current, and n denotes the discretized time-step index. Eq. 2.27 forms up the basic $\mathcal{G}$ or $\mathcal{F}$ with the global circuit solution, where the capacitor voltages $\boldsymbol{v}_c$ and arm current i_{arm} are the state variables required in PiT iterations. Since i_{arm} can be inferred from MMC nodal voltages, the final state vector of ideal switch MMC is $\boldsymbol{U} = \{v_{dc}^p, v_{dc}^n, v_a, v_b, v_c, \boldsymbol{v}_c\}$.

Transient Curve-Fitting Model The transient curve-fitting model provides device-level information while also maintaining higher computational

efficiency than the nonlinear physical model. The datasheet-driven model contains two stages [3]:

(1) *Steady-State*: it is represented by a resistor similar to the ideal switch model with its value being determined by static $i_C - v_{CE}$ curves, which can be expressed by

$$\begin{aligned} r_s\left(i_C, v_g, T_{vj}\right) &= \frac{V_{CE}}{i_C} \\ &= a_1\left(i_C, v_g, T_{vj}\right) + a_2\left(i_C, v_g, T_{vj}\right) \cdot I_C^{-1}, \end{aligned} \tag{2.28}$$

where i_C is the collector current, a_1 and a_2 are coefficients dependent on factors such as the gate voltage v_g, and junction temperature T_{vj}.

(2) *Transient-State*: it models the turn-on and turn-off transient behaviors of collector current i_C and v_{CE}. The rising and falling time t_r, t_f are, based on Stone–Weierstrass theorem, approximated by a polynomial function

$$\begin{aligned} t_{r,f}\left(s_1, s_2, s_3\right) = k_0 \cdot \prod_{i=1}^{2} s_i + \\ \sum_{i,j=1,2,3}^{i \neq j} k_i s_i s_j + + \sum_{i=1}^{3} b_i s_i + b_0, \end{aligned} \tag{2.29}$$

where the variable s_i represents one of the 3 factors such as i_C, T_{vj}, and R_g provided in the datasheet; k_i, b_i are constants. The turn-off and turn-on transients are modelled by a virtual RC circuit which has $\tau = r * C_g, C_g = 1nF$. The τ is related to $t_{r,f}$ and the virtual capacitor voltage v_{Cg} must be initialized according to the transient type. The turn-on transient is triggered by high-level (rising edge) control signals which charges the v_{Cg} to 1V; on the other hand, the turn-off transient is triggered by low-level (falling edge) control signals and the RC circuit discharges from v_{Cg} to 0. During the RC circuit calculation, a current source is computed for generating transient i_C across the collector and emitter poles of the IGBT, which is computed by

$$i_{Cs}(n) = \begin{cases} 0, & \left(v_{Cg} = 1\right), \\ i_{Cs}(n-1) + k_1 \cdot \Delta t/\tau, & \left(v_{Cg} \leq 1 - e^{\frac{-2tr}{\tau}}\right), \\ \left(i_{Cs}(n-1) - 1\right) \cdot e^{\frac{-t}{k_2}} + i_C, & \left(v_{Cg} > 1 - e^{\frac{-2t_f}{\tau}}\right), \end{cases} \tag{2.30}$$

where i_{Cs} is the current source, k_1 and k_2 denote the rise and fall rates of the current, and i_C is the steady-state current. In this work the ABB®

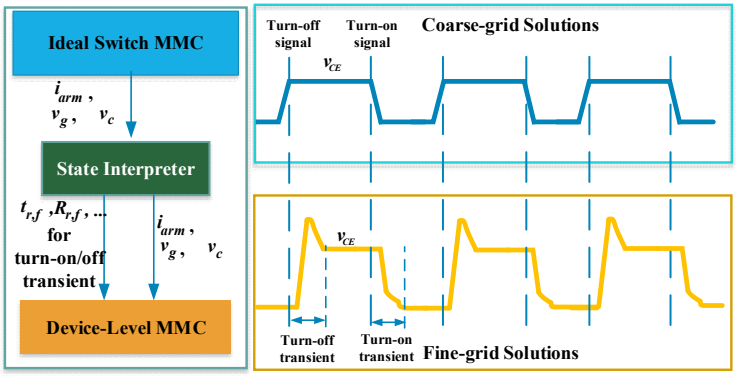

Figure 2.11: State interpretation from the coarse-grid ideal switch model to the fine-grid device-level model.

5SNA2000K450300 StakPak IGBT module is selected for device-level HBSMs [80].

Although it can perform PiT computation by registering all internal states into the iteration, the model causes great difficulties to the PiT iteration due to these internal transient states and time-step limitations, making the PiT method slower than the sequential one. Therefore, a hybrid PiT solution is proposed by using the ideal switch model as a coarse predictor. In this case, the additional state variable from the curve-fitting HBSM is the gate signals $\boldsymbol{v}_g$ from the last time-step, which can determine the next IGBT stage (turn-on or turn-off) along with the new control signals.

2.6 Parallel-in-Time Implementations

The synchronization of controllers with discrete switching events plays a significant role in the algorithm. If the controller uses the simulation time-step of their worker, the fine-grid and coarse-grid results differ greatly due to the different firing signals generated by controllers under different time-steps. In practice, even a tiny difference can produce inconsistent results between coarse-grid and fine-grid. The solution is to define a fixed sampling time for the controller in both grids, and it is better to be the same as the coarse-grid time-step.

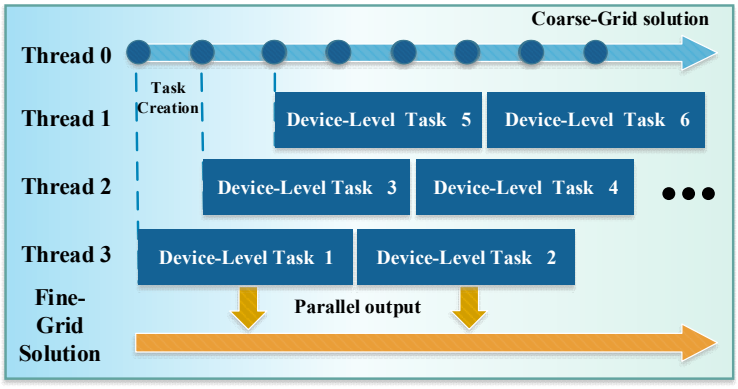

Figure 2.12: OpenMP® task-based parallelism for hybrid PiT method for MMC.

This method works fine with the ideal switch model. However, the iteration process becomes much slower when the MMC scale increases. As more state variables are involved in the iteration, the algorithm becomes harder to converge, and the overhead increases a lot so that the speed-up cannot be achieved. This becomes worse with the more complex device-level HBSM model. Thus, a hybrid PiT model is proposed.

Hybrid PiT Model The curving-fitting model reflects the behaviour of the IGBT devices by using simple RC circuits to generate device-level turn-on and turn-off transients. It is almost equivalent to the ideal switch model in the steady-state. The controller events can be captured on the coarse-grid so that the transient-state will function normally within the fine-grid just like the serial implementation. The core PiT concept of the Parareal is to find a computationally cheap coarse predictor to make an initial guess, so combining the system and device-level model may be a good solution, which is considered a hybrid model. A similar approach can be found in [81] for FPGA real-time simulation without PiT.

The basic concept of the hybrid PiT model is shown in Fig. 2.11. The coarse-grid arm current i_{arm}, gate signals $\boldsymbol{v}_g$, capacitor voltages $\boldsymbol{v}_c$ obtained from ideal switch models are passed to state interpreter first; the state interpreter determines the device-level IGBT states and initial the variables for the detailed transient; finally, the fine-grid device-level curve-fitting model

produces the transients in parallel. The control system only operates on the coarse-grid so there is no different behaviour between coarse-grid and fine-grid control signals, making the synchronization between coarse-grid and fine-grid much easier. From the experimental results, the ideal switch model produces accurate enough system-level results so it is possible to skip the PiT iteration process for extreme performance. The PiT algorithm is generally a trade-off between accuracy and performance, so if some fault transients require high accuracy or cause diverge, the algorithm can always use a smaller time-step or revert to sequential to overcome these difficulties and go back to PiT in the steady-state.

The hybrid PiT implementation is based on OpenMP® API. OpenMP® pragmas are directly compiled by the C/C++ compiler which masks the complexity to call system-specific API functions. Moreover, the OpenMP® API provides many useful functionalities such as thread-pooling, load-balancing, tasking and even heterogeneous hardware off-loading in the OpenMP 5.0 standard, which are extremely useful for parallel computing [82]. In Fig.2.12, the coarse-grid predictor launches a fine-grid parallel task when a solution is ready, which is implemented by the OpenMP® tasking feature. This task-based parallel scheme can hide the states translating latency and benefit from the workload balancing by the task scheduler. A fixed-size block of memory is assigned to each task for the outputs.; each task writes to the final result vector in parallel and overrides the original coarse-grid time points.

TLM-based PiT+PiS Method Different kinds of elements exist in power systems, while not all of them currently have a PiT implementation. Transmission lines modelled based on travelling wave theory bring delay differential equations (DDEs) to the systems of EMT simulation [83], which needs extra treatments in PiT methods. The transmission lines bring DDE challenges, but it also makes it possible to divide a large PiT system into decoupled PiT subsystems.

Fig. 2.13 shows the overview of a PiT+PiS simulation architecture. Because of the usage of TLMs and PiT adapters, the PiT and traditional PiS

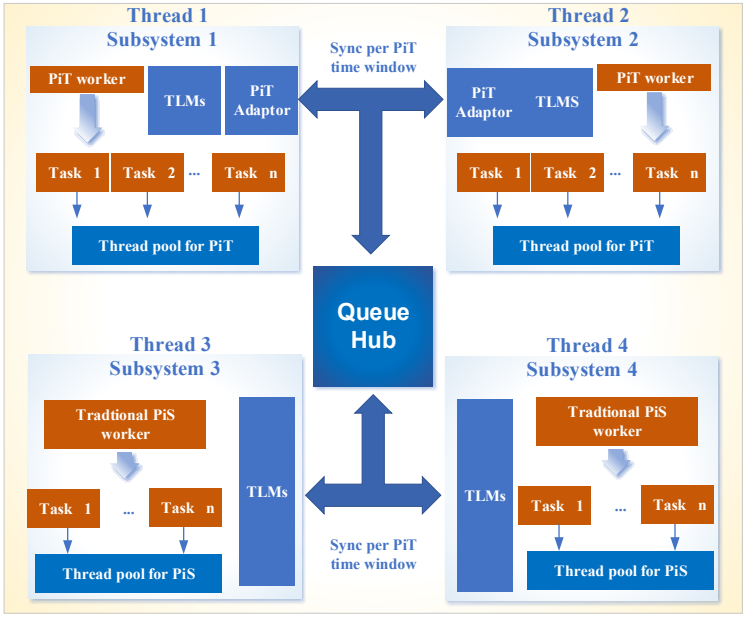

Figure 2.13: PiT+PiS simulation architecture.

methods can be used in subsystems without modifications. The only difference is that all subsystems should synchronize at the minimal time window required by PiT systems.

The transmission line elements connect two systems via a sending-end and a receiving-end queue, which are usually implemented by ring buffers. All queues are managed by a global queue hub. The queue hub is a hash map in which all transmission line elements can look up their queue objects by the unique designated names in a string.

The coarse-grid and fine-grid line models in the parallel-in-time implementation must use the same fine-grid history data to get correct solutions due to the TLM limitation. Besides, the iteration process requires restarting the simulation. Therefore, a PiT adapter is used to decouple the queue hub from PiT TLM and traditional TLM. The PiT adapter is a simple memory buffer for PiT systems to repeatedly read the history vector from other systems so that the iteration process can proceed normally. The size of time windows is limited by the propagation delay and the memory size of buffers should be pre-determined according to the system configurations.

In the conventional implementation, a pair of transmission lines will send

and receive one packet of history data via the queues at each simulation step. For PiT implementation, it is necessary to synchronize per time window. It also has a PiT adapter as a buffer between conventional and PiT line models since the PiT transmission line models may have different interpretation schemes.

2.7 Results and Discussion

Two cases are presented to evaluate the accuracy and performance of proposed methods. The performance is evaluated by the theoretical speed-up, actual speed-up, and parallel efficiency. The theoretical speed-up depends on recording function calls to coarse-grid and fine-grid functions:

$$S_{theor} = \frac{C_{coarse} + C_{fine} * R_{fine}}{C_{seq}}, \tag{2.31}$$

where S_{theor} is the theoretical speed-up, C_{coarse} is the number of function calls to coarse-grid, C_{fine} is the function calls to fine-grid, R_{fine} is the number of simulation steps in each fine-grid function call, and C_{seq} is the number of simulation steps of the traditional sequential program. By using discrete steps to compute the speed-up, all software overhead is neglected. Therefore, it is the best performance that the Parareal algorithm can achieve. Since the numbers of simulation steps of different models are not comparable, only the theoretical speed-up of PiT for the ideal switch model is evaluated.

The actual speed-up is the PiT execution time divided by sequential execution time, and the parallel efficiency is computed by

$$E_{par} = \frac{T_s}{N * T_p} = \frac{S}{N}, \tag{2.32}$$

where T_s is the execution time of sequential programs, T_p is the execution time of parallel programs, N is the number of threads and S is the actual speed-up.

CASE 1: Single MMC The Cm-A1 MMC station from DC System (DCS) 1 of CIGRÉ B4 DC grid is utilized for the testing of both ideal and transient curve-fitting switch models. Resistors are connected between each DC line

and ground to generate a rated load of 800MW. All submodules are computed individually in the MMC.

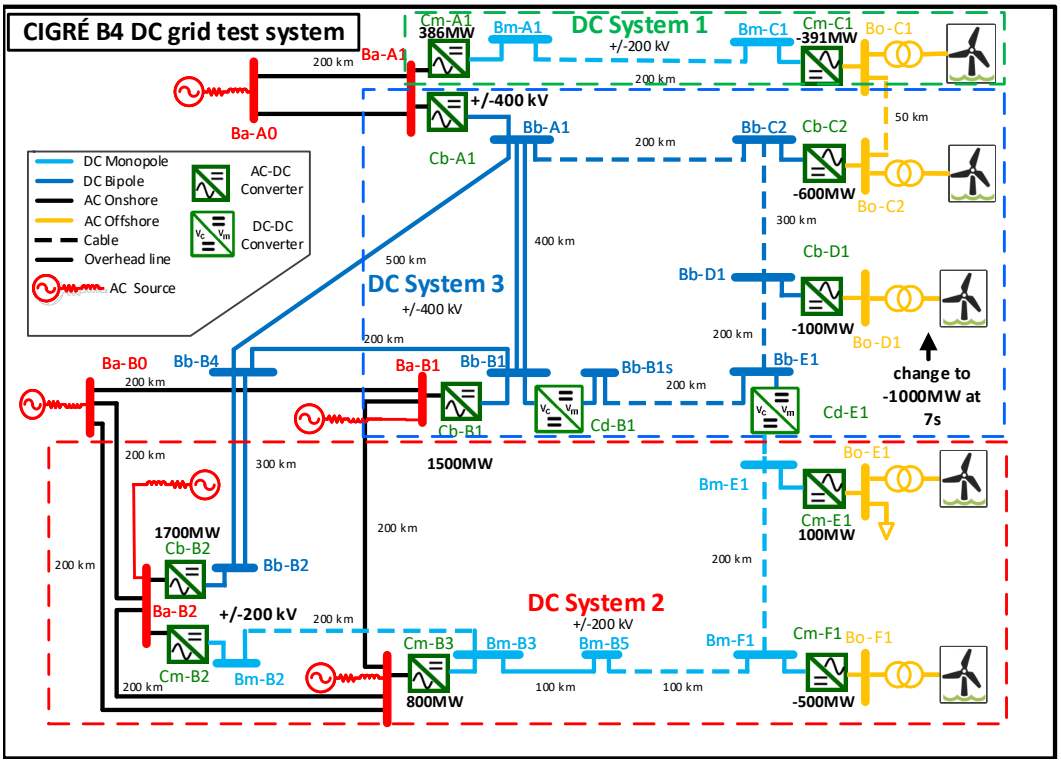

Figure 2.14: CIGRÉ HVDC grid test system for the case study.

The results are shown in Fig. 2.15. The *Fine* results were produced from the ideal switch model with $\delta t = 1\mu s$, and the *Coarse* results were from the ideal switch model with $\Delta t = 1\mu s$, the *PiT* results were from hybrid MMC model. For the system-level solutions, the relative error of the hybrid PiT algorithm is smaller than 0.1%, while the coarse solution ($\Delta t = 50\mu$s) has an error of around 2%. This means the hybrid method achieved the accuracy of fine-grid small time-step while the computation time is reduced significantly by PiT solution, as compared with the ideal switch model under coarse time-step, the MMC currents of the proposed method are more accurate (error <0.5%) than the *Fine* solution attributing to accurate initial values, in addition to a small truncation error accompanied by the applied time-step.

The hybrid PiT method not only has the same system-level results of accurate solutions but also produces device-level waveform, which is shown in Fig. 2.15 (c) and (g). In these two figures, the coarse ideal-switch solutions are

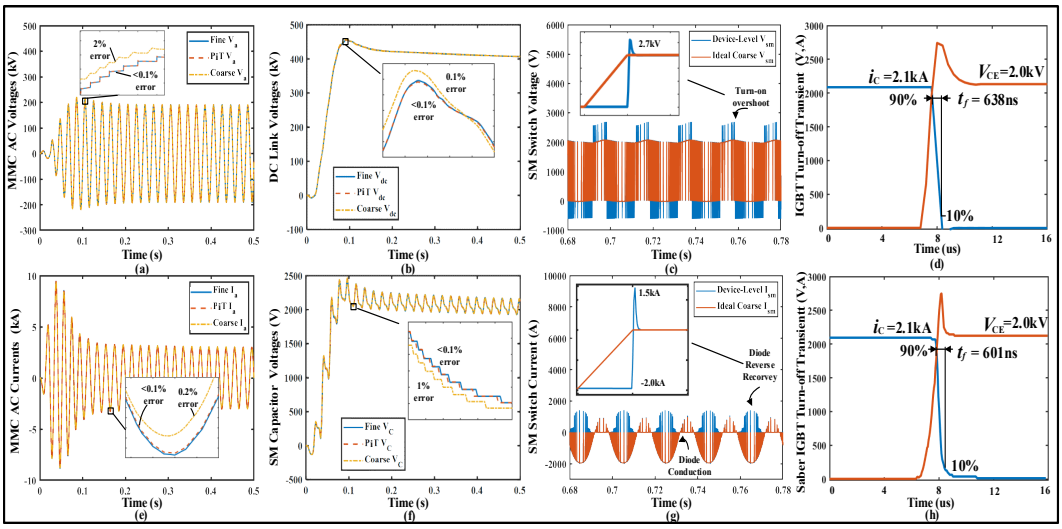

Figure 2.15: Simulation results of Case 1 with PiT and serial methods (a):Three-phase voltages of MMC (b):DC link voltages (c):Voltages across the IGBT S_2 of SM-0 (d):IGBT turn-off transient from PiT method (e):Three-phase currents of MMC (f):Capacitor voltages of SM-0 (g):Currents through the IGBT S_2 of SM-0 (h):IGBT turn-off transient from fourth-order IGBT model igbt1_3x in SaberRD®

Table 2.4: Performance Comparison of Ideal Switch Parareal Algorithm With Different MMC Scales Over Simulation Duration=1s, Thread Number=6, $\Delta t = 50\mu s$ and $\Delta t = 1\mu s$.

SM Num. N_{sm}	Parareal (s)	Traditional (s)	Theoretical speed-up	Actual speed-up	Average iterations
8	0.31	0.39	2.33	1.28	2.26
16	0.42	0.55	2.20	1.29	2.41
32	0.54	0.67	2.06	1.24	2.57
64	0.92	1.05	1.93	1.13	2.77
100	1.40	1.60	1.88	1.14	2.84
200	2.84	2.74	1.84	0.96	2.91

step-wise trapezoidal waveform due to the large time-step, while the device-level solutions reflect the voltage overshoot and diode reverse recovery. The hybrid model IGBT turn-off transient results in Fig. 2.15 (d) are compared to the Fig. 2.15 (h), which is the nonlinear behavioral model simulation in SaberRD® conducted under a variable time-step up to 10ns. Despite the curve-fitting model is computed with a time-step of 1μs, its maximum voltage and transient duration are close to SaberRD®.

Table 2.5: Performance Comparison of hybrid Algorithm With Different MMC Scales Over Simulation Duration = 1s, Thread Number = 6, $\Delta t = 50\mu s$ and $\Delta t = 1\mu s$.

SM Number N_{sm}	Hybrid PiT (s)	Curve-fitting Model (s)	Speed-up	Parallel efficiency
8	0.168	0.803	4.78	0.80
16	0.267	1.511	5.65	0.94
32	0.471	2.695	5.72	0.96
64	0.872	5.062	5.80	0.97
100	1.714	7.662	4.47	0.75
200	3.491	15.50	4.44	0.74

Table 2.6: Performance Test of Hybrid Algorithm With Different Number of Threads Over $N_{sm} = 200$, Simulation Duration = 1s, $\Delta t = 50\mu s$ and $\Delta t = 1\mu s$.

Threads	Time (s)	Speed-up	Effiency
48	0.527	30.42	0.63
24	0.950	16.87	0.70
12	1.838	8.72	0.73
6	3.612	4.44	0.74
4	5.478	2.93	0.73
2	10.998	1.46	0.73

The performance of PiT methods is compared in Table 2.4 and 2.5, where the speed-up is defined as the proposed parallel execution time over that of the conventional method, and efficiency is calculated as speed-up divided by the number of threads. Table 2.4 is the traditional Parareal implementation with the ideal switch model. With SM numbers lower than 100 per arm, the Parareal program can be 110-120% faster than the serial program. When the SM number increases to 200, the Parareal method loses speed-up. Compared to the highly optimized serial program, although the Parareal gets accurate results, its iteration overhead is so large that it is difficult to get a reasonable speed-up with a large SM number.

In Table 2.5, the hybrid model gets much better performance with a trade-off of accuracy. The parallel efficiency is greater than 70% in all cases. When the number of SMs N_{sm} is between 16-64, almost full parallel efficiency is obtained. This can be inferred from Fig. 2.12 as when the conventional program

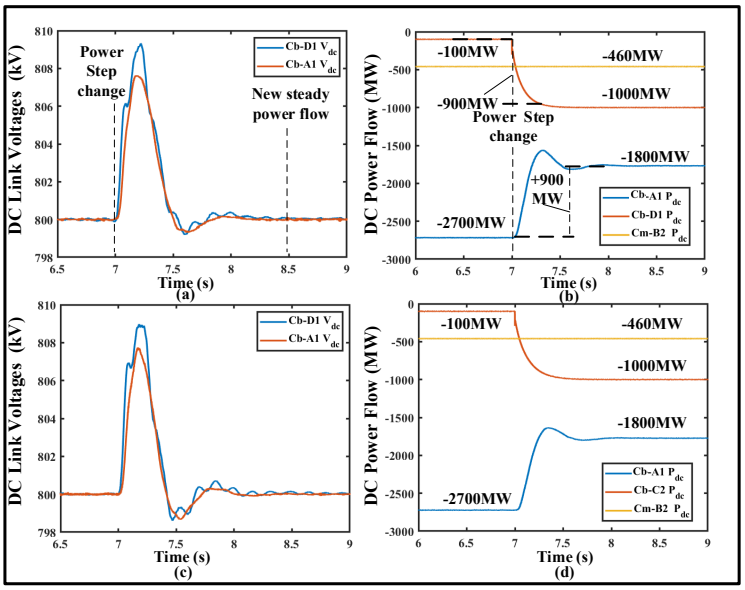

Figure 2.16: Simulation results of CIGRÉ B4 DC grid test system (a):DC-link voltages when power step-change at 7s from the PiT+PiS program (b):Real power flow at MMC stations from the PiT+PiS program (c):DC-link voltages from PSCAD/EMTDC® (d):Real power flow from PSCAD/EMTDC®

finishes the solution whose length equals Task 1, the PiT method has almost finished 6 tasks since the coarse prediction in Thread 0 is much faster than the conventional model, and the Thread 0 can also handle tasks when it is idle. If the predictor is not fast enough or the task executes much longer than the conventional implementation, which is similar to the $N_{sm} = 8$ or 200, the efficiency drops. Thus, this method sacrificed a little accuracy but gets a huge performance boost.

Since the single MMC station has no limit on the time window, the thread number can go up to a maximum of 48. Table 2.6 shows the multi-thread performance of the hybrid algorithm with a 201-level MMC. High parallel efficiency greater than 60% is obtained in all test cases. Notice that the simulation for 1s only takes 0.527s when using 48 threads (30x speed-up) and $\delta t = 1\mu s$.

CASE 2: CIGRÉ HVDC Grid The full-scale CIGRÉ HVDC grid test system is used to verify the connection among PiT systems. DC-DC converter Cd-B1 is bypassed and Cd-E1 is disconnected following the guide of the CIGRÉ B4 test case. All MMC parameters including controller set-points are set

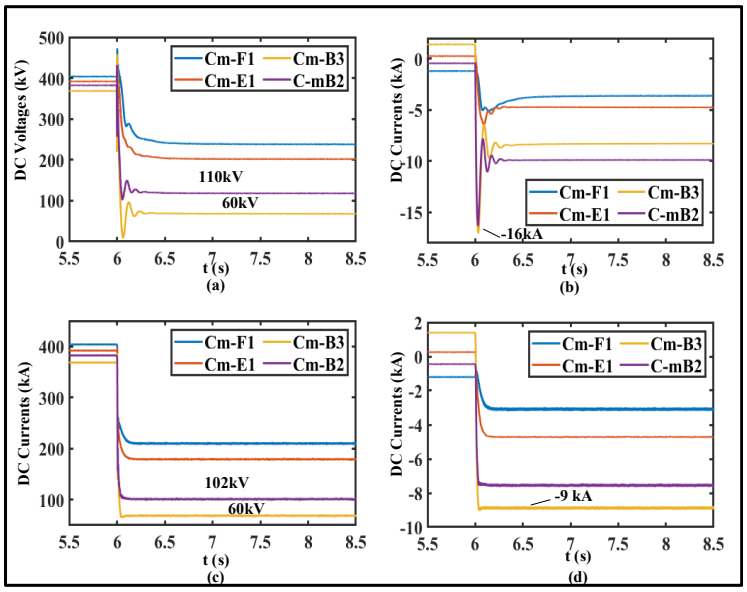

Figure 2.17: Simulation results of a DC line fault in CIGRÉ B4 DCS 2 system (a):MMC DC voltages without blocked SMs (b):MMC DC currents without blocked SMs (c):MMC DC voltages with blocked SMs (d):MMC DC currents with blocked SMs

Table 2.7: Performance Comparison of TLM PiS , PiT+ PiS and Serial Program, Over Simulation Duration=1s, $\Delta t = 50\mu s$ and $\Delta t = 1\mu s$.

PiS single level parallel computing				PiT+PiS nested parallel computing			
Thread	Time (s)	Speed-up	Eff.	Thread	Time (s)	Speed-up	Eff.
1	294.60	1.00	1.00	4	105.21	2.81	0.70
2	171.57	1.72	0.86	8	69.79	4.22	0.53
4	86.55	3.40	0.85	16	34.53	8.53	0.53
6	58.38	5.05	0.84	24	15.60	18.89	0.79
8	58.85	5.01	0.63	32	12.92	22.80	0.71
11	30.57	9.64	0.88	44	14.12	20.86	0.47
16	29.49	9.99	0.62	64	12.70	23.20	0.36
32	29.88	9.86	0.31	128	12.69	23.21	0.18
48	29.80	9.88	0.21	192	13.58	21.70	0.11

according to the standard power flow as shown in Fig. 2.14, except Cb-D1, which has only 100MW generating power at the beginning. The Cb-D1 power will change to 1000MW at 7s, and the standard power flow should be observed at Cb-A1. All MMCs have 200 curve-fitting device-level HBSMs per arm.

The results are shown in Fig. 2.16. Fig. 2.16 (b) shows the power flow at Cb-A1 and Cb-D1. As the power output of Cb-D1 increases, the Cb-A1

returns the same amount of power to the AC system; the final power flow is -1800MW while the standard power flow file is around -2000MW, which is a little different. This is because the DC-DC station Cd-E1 is supposed to transmit 300MW power from DCS3 to DCS2 in the standard case. In the simplified case, Cd-E1 is disconnected so the surplus 300MW power is supplied by Cm-B2, and with the surplus real power, the Cb-A1's power generation decreases.

Another scenario for Case 2 is a DC fault that occurs at 6s in DCS 2. The phase-to-phase fault resistance is 1Ω, which is located in the middle of Bm-B2 and Bm-B3. The results are shown in Fig. 2.17; the Fig. 2.17 (a) and (b) are DC voltages and currents respectively of MMCs in DCS 2 with no change to the SM states. The Fig. 2.17 (c) and (d) are the results of MMCs with SMs changed to blocking state 100μs after the phase-to-phase fault. The Cm-B2 and Cm-B3 have a larger impact on voltages and currents since they are closer to the fault location. Comparing the blocked SMs to the unblocked SMs, the fault voltages are similar but their transients become different as the blocked version has smaller oscillations. The currents are more different and the maximum fault current of blocked SMs is about half of the unblocked SMs' maximum fault current. The fault current reaches the peak value within 50ms in both cases, which indicates that faster protection equipment is required to effectively protect the power system from the potential damage of DC faults.

Only hybrid PiT program performance tests are presented. For the PiT program, 2.81x speed-up and 70% efficiency can be obtained with four threads. The performance test on the CIGRÉ system is different from the previous ones. Since multiple MMC stations can be executed in parallel with high efficiency, it is worthy to check the performance of PiT+PiS against the traditional TLM decoupled PiS method. However, it creates two levels of parallelism with a nested relationship, which brings extra complexity and overhead. In this test, a number of PiS threads are assigned to the first level of parallelism; each PiS thread launches its own team of 4 threads for PiT tasks at the second level; different teams cannot communicate with each other.

Results are given in Table 2.7. The PiS method achieves around 80%

efficiency when the thread number is equal to or less than 11 with an exception of $N = 8$. When $N = 8$, it may be affected by the load-imbalance so that the efficiency is lower than the balanced case $N = 11$. When the thread number is greater than 11, the speed-up stops at 10x since there are only 11 MMC stations. For the PiT+PiS program, the speed-up and efficiency are generally lower than pure PiS or PiT cases due to the overhead brought by the nested parallelism. However, it extends the boundary of overall speed-up and can utilize more threads to accelerate the simulation. Therefore, greater than 20x speed-up is obtained with $N > 24$.

2.8 Summary

A component-based simulation class architecture is proposed to handle different models in power systems, which delivers high flexibility and scalability to implement the system-level PiT algorithm. Major challenges to handle the DDEs brought by transmission line models in the PiT algorithm are analyzed and a modified model implementation is proposed to solve the problem. Using the proposed circuit solver class as basic workers, the PiT algorithm based on the Parareal is implemented using object-oriented C++. The case study shows accurate results compared to the sequential program, while better parallel speed-up and efficiency than the MKL parallel LU implementation are obtained, showing the great potential of accelerating power system EMT simulation. The performance test also shows the system's time-domain characteristics determine the speed-up of Parareal algorithm, especially the transmission line delay in power system EMT simulation.

EMT simulation of MTDC grids especially with device-level models for MMC converters is time-consuming not only due to the accuracy constraint but also due to the excessive computational burden. this work investigated a joint parallel scheme that takes both advantages of traditional TLM-based spatial and Parareal-based temporal parallel techniques. To realize the device-level IGBT transient simulation under PiT, a hybrid PiT method that utilizes ideal switch as the coarse-grid predictor and curve-fitting IGBT model

as fine-grid corrector is proposed and implemented on multi-core CPU using OpenMP® tasking to achieve maximum efficiency. A TLM propagation-delay-based method is integrated into the PiT systems to enable flexibility regarding forming an MTDC grid. Test results show a 30x speed-up of the hybrid PiT model for device-level simulation of a 201-level three-phase MMC with 48 threads, with an error of less than 2%. High parallel efficiency from 60% to 97% is achieved with the proposed hybrid model. For the CIGRÉ HVDC grid test system, the PiT+PiS method shows a 20x greater speed-up than the pure PiT or PiS method. The case studies on a single MMC and CIGRÉ B4 system show a significant speed-up of the proposed methods and indicate great potentials for hybrid PiT+PiS methods for device-level MTDC EMT simulation. Future applications can be extended to nonlinear physical power electronic simulation, the power system with renewable energy sources, and other applications with complex device-level transients.

Chapter 3

Hybrid Parallel-in-Time-and-Space Transient Stability Simulation of Large-Scale AC/DC Grids

3.1 Introduction

[1] This chapter proposes a hybrid PiT+PiS AC/DC TS-EMT co-simulation method which thoroughly exploits parallelism to expedite the simulation of modern power systems on heterogeneous CPU-GPU hardware platform. It analyzes the theoretical speedup analysis of PiT and PiT+PiS methods implemented the PiT+PiS TS-EMT co-simulation algorithm on heterogeneous CPU-GPU hardware platform. The advanced asynchronous multi-stream design and unified GPU memory are utilized to achieve high parallel efficiency and the advantages are proved by a large-scale TS-EMT AC-DC co-simulation system.

Modern power systems are increasingly complex due to the continuous integration of power electronic facilities such as the high voltage direct current (HVDC) links into transmission and distribution networks. Many HVDC projects are constructed or planned worldwide to integrate more clean energy from wind farms and PV stations, such as Changji-GuQuan UHVDC project

[1]This work has been published: **T. Cheng**, N. Lin and V. Dinavahi, "Hybrid parallel-in-time-and-space transient stability simulation of large-scale AC/DC grids," *IEEE Trans. Power Syst.*, vol. 37, no. 6, pp. 4709-4719, Nov. 2022, doi: 10.1109/TPWRS.2022.3153450.

in China [84], Dolwin 5 project in Europe [85], and TransWest project in the USA [86]. Using a typical step size of a few milliseconds, the TS analysis plays an important role in the planning, design, and operation of a modern power grid from a system point of view. It, however, is a positive-sequence-based analytical method that naturally falls short of complicated EMT details of power electronic devices in the microsecond-level or below. The TS-EMT co-simulation methodology which properly features both system-level and equipment-level power system phenomena is favored in hybrid AC/DC grid study. A consequently incurred rise of computational burden may extend the simulation duration, especially considering that a dramatic expansion of a future AC/DC power system as a result of incorporating more components is expected.

To handle the increasing scale and complexities, new acceleration techniques for TS and EMT simulation programs are desired. TS acceleration methods based on parallel processing algorithms for multi-core CPUs and many-core GPUs have shown a decent efficiency and have been well investigated in AC power grid studies [6], [22]–[24], and heterogeneous CPU-GPU computing architecture for AC-DC grid TS-EMT co-simulation has recently been proposed [25]–[27], [87], while the threads concurrency of these methods is dominantly contributed by parallel-in-space (PiS) strategies. The PiT solutions were also investigated from a different perspective of parallel processing[18]. The very early version of PiT for solving TS problems was mainly based on Jacobi-decomposition [28], [29], which aimed to solve multiple continuous steps iteratively so that the parallelism was achieved by assigning a single step to parallel threads. In the 1990s, most results were obtained from the virtual parallel machine because of a scarcity of multi-core CPUs, which limited further explorations in this area. The Parareal algorithm has emerged in many research areas [21], [88] where it exhibited efficacy [30]. It was introduced to solve TS simulation problems by decomposing the initial value problem into many sub-intervals [19], and has a better efficiency compared to its predecessors. These works mainly focused on PiT algorithms and potential comprehensive parallelism by considering PiS methods simultaneously is yet

to be carried out. For example, a four times speedup was obtained in computing the IEEE 39-bus system [19] with 470 cores, and even a huge number of cores were used to solve a large-scale power system, the parallel efficiency was below 20% [31], which is still not as satisfactory as PiS methods.

3.2 Multi-Mass Torsional Shaft Generator Model

The generator equations comprise of three components, i.e., the synchronous machine, the mechanical multi-mass torsional shaft, and the control system, which together form a 17th-order model.

As shown in Fig. 3.1, the synchronous machine includes two windings on d-axis and two damping windings on q-axis, which is classified as Model 2.2 in IEEE Std 1110-2019 [89]. The machine model can be expressed by differential equations as [90]:

$$\begin{aligned}
\dot{\Psi}_{fd}(t) &= \omega_R \cdot [e_{fd}(t) - R_{fd} i_{fd}(t)] \\
\dot{\Psi}_{1d}(t) &= -\omega_R \cdot R_{1d} i_{1d}(t) \\
\dot{\Psi}_{1q}(t) &= -\omega_R \cdot R_{1q} i_{1q}(t) \\
\dot{\Psi}_{2q}(t) &= -\omega_R \cdot R_{2q} i_{2q}(t),
\end{aligned} \tag{3.1}$$

where Ψ_{fd} and e_{fd} are flux linkage and field voltage of the field winding, Ψ_{1d}, Ψ_{1q}, Ψ_{2q} are the flux linkages of direct and quadrature axis windings; R_{1d}, R_{1q} and R_{2q} are the resistance of direct and quadrature axis windings; ω_R is the rated rotating speed. All the quantities are using the per unit system defined in [90] except the time t since the time domain simulation use seconds for the time unit. The currents in the dq frame are coupled with external power system equations as the stator currents must be obtained. To simplify the computation, an iterative method is used to handle the coupling [23].

The mechanical shaft, on the other hand, provides basic rotor equations for the machine:

$$\begin{aligned}
\dot{\delta}(t) &= \omega_R \cdot \Delta\omega(t), \\
\Delta\dot{\omega}(t) &= \frac{1}{2H} \left[T_m(t) - T_e(t) - D \cdot \Delta\omega(t) \right].
\end{aligned} \tag{3.2}$$

Since the synchronous generator is connected to the turbine, the four-mass-turbine shaft model is used, as shown in Fig. 3.2, where δ_n means the relative

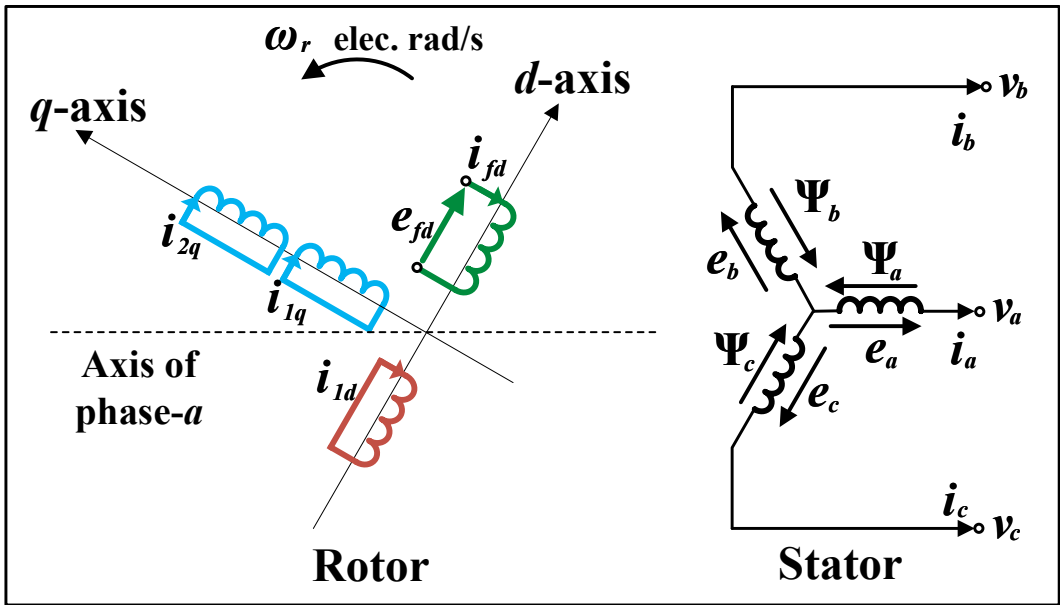

Figure 3.1: Rotor and stator circuits of a Model 2.2 synchronous machine.

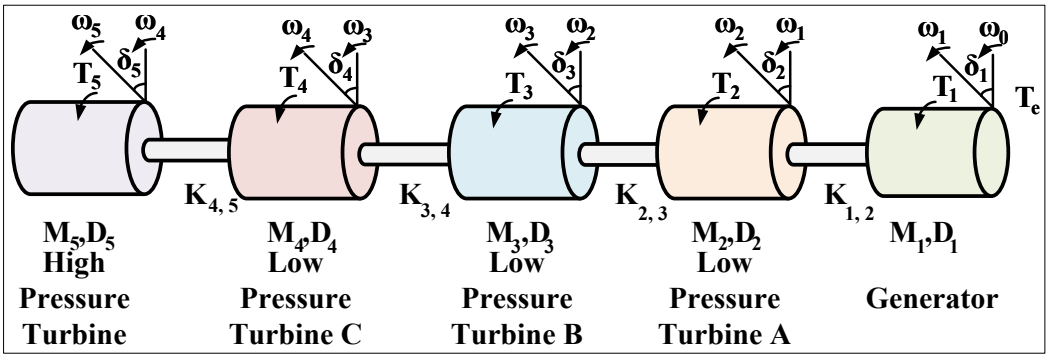

Figure 3.2: Multi-mass model for mechanical side of the generator.

rotor angle of each turbine, and specifically, δ_1 is the generator rotor angle. The stiffness coefficient between two neighboring masses is represented by parameter K, e.g., $K_{4,5}$ represents the coefficient between Mass5 and Mass4. T_n, D_n and H_n refer to the torque, damping factor and inertia constant of each torsional shaft, respectively, as given by:

$$\begin{aligned} &\Delta\dot{\omega}_n(t) = \\ &\frac{\left[T_n + K_n^{n+1}(\delta_{n+1}(t) - \delta_n(t)) - K_{n-1}^n(\delta_n(t) - \delta_{n-1}(t)) - D_n\Delta\omega_n(t)\right]}{2H_n}, \\ &\dot{\delta}_n(t) = \omega_R \cdot \Delta\omega_{n(t)}, n = 2, 3, 4, 5. \end{aligned} \tag{3.3}$$

The control system includes PSS, excitation, and AVR systems, which is classified as ST1A model in [91]:

$$
\begin{aligned}
\dot{v}_1(t) &= \frac{1}{\tau_R} \cdot [|\boldsymbol{v}_{dq}| - v_1(t)] \\
\dot{v}_2(t) &= K_{stab} \cdot \Delta\dot{\omega}(t) - \frac{1}{\tau_\omega} v_2(t) \\
\dot{v}_3(t) &= \frac{1}{\tau_2} \cdot [\tau_1 \dot{v}_2(t) + v_2(t) - v_3(t)] ,
\end{aligned}
\tag{3.4}
$$

where τ_R, τ_ω, τ_1, τ_2, and K_{stab} are constants, v_1, v_2, v_3 are state variables. The generator equations and power system network equations constitute the differential-algebraic equations (DAEs) of transient stability simulation, which can be expressed as

$$
\begin{aligned}
\dot{\boldsymbol{x}} &= f(\boldsymbol{x}, \boldsymbol{u}), \quad g(\boldsymbol{x}, \boldsymbol{u}) = 0, \\
\boldsymbol{x}_0 &= \boldsymbol{x}(t_0), \qquad \boldsymbol{i}_0 = \boldsymbol{i}(t_0),
\end{aligned}
\tag{3.5}
$$

where $\boldsymbol{x} = \{\Psi_{fd}, \Psi_{1d}, \Psi_{1q}, \Psi_{2q}, \delta_1, \Delta\omega_1, \delta_2, \Delta\omega_2, \delta_3, \Delta\omega_3,$
$\delta_4, \Delta\omega_4, \delta_5, \Delta\omega_5, v_1, v_2, v_3\}$ is the generator state vector , $\boldsymbol{u} = \{i_{1d}, i_{1q}, i_{2q}, e_{fd}, i_{fd}, \boldsymbol{v}_{dq}\}$ is the vector of system inputs; f is the vector function of equations (3.1)-(3.4); g is the algebraic equation to solve the input vector $\boldsymbol{u}$.

In addition, the stator equations [90], [92] are necessary to solve the interface voltages and currents along with the external grid, making the simulation a DAE problem. The generator's variables are in d- and q-axis so Park transformation is involved in solving the DAE [93], which makes it a nonlinear problem. As shown in Fig. 2, a partitioned iterative method [23] is used to solve the DAE, which decomposes generators from the main circuits.

With implicit Trapezoidal Rule, an individual generator has the following discretized equation

$$
\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \frac{1}{2} h(\boldsymbol{f}(\boldsymbol{x}_{n+1}, \boldsymbol{u}_{n+1}) + \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{u}_n)),
\tag{3.6}
$$

where n indicates the number of discrete steps, h indicates the time-step. Since solving such an implicit equation requires f_{n+1} at x_{n+1} and u_{n+1}, implicit methods requires the Jacobian matrix $\boldsymbol{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$. The generator equation can be expressed by a state-space form:

$$
\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = A\boldsymbol{x} + B\boldsymbol{u},
\tag{3.7}
$$

-

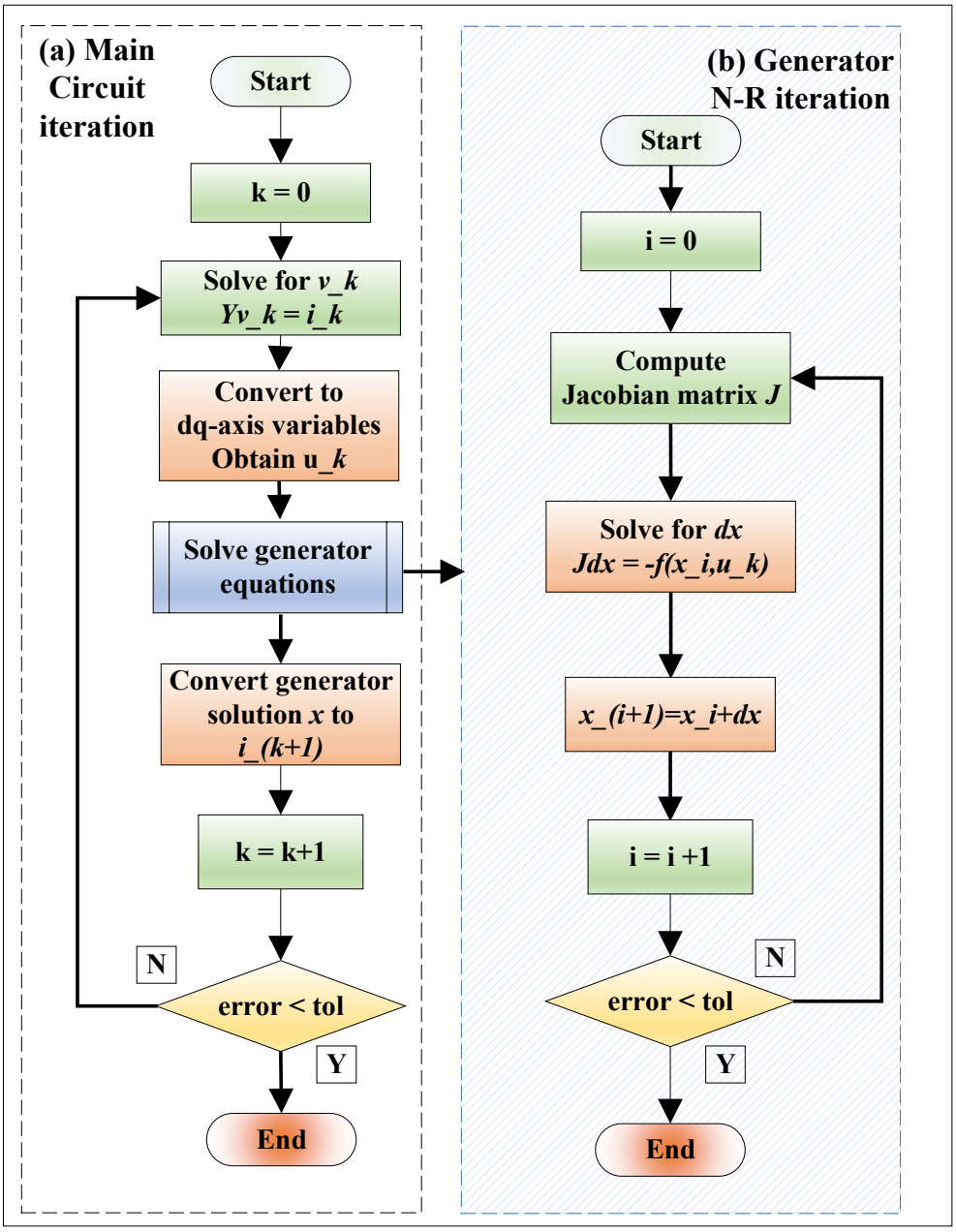

Figure 3.3: Transient stability simulation process: (a) loop for solving voltages in main circuit; (b) loop for solving generator state variables.

where A and B are coefficient matrices of Equation (3.1-3.4). Then the equation to solve x_{n+1} can be expressed by

$$(E - \frac{h}{2}A)\boldsymbol{x}_{n+1} - (E + \frac{h}{2}A)x_n + \frac{h}{2}B(\boldsymbol{u}_{n+1} + \boldsymbol{u}_n) = 0 \tag{3.8}$$

where $(E - \frac{h}{2}A)$ is the Jacobian matrix. If f_{n+1} is nonlinear, A is no longer constant so that the Newton-Raphson iteration is required. Otherwise, it can be solved without local iteration [94].

3.2.1 Theoretical Speedup Analysis

Assuming a system with a fixed workload of $n \cdot w$, where n indicates the total fine-grid time-steps of the simulation and w is the single-step execution time of the DAE solution, and the same integration method for the fine-grid and coarse-grid, according to Amdahl's law [95], the speedup of the PiT algorithm can be expressed by

$$S_{pit}(p) = \frac{n \cdot w}{((I+1) \cdot p \cdot w + I \cdot m \cdot w)} = \frac{1}{((I+1)/m + I/p)}, \tag{3.9}$$

where S_{pit} is the speedup of Parareal, m is the steps of fine-grid sub-intervals, I is the iteration number and p is the number of PiT processors, $n = mp$. The number of parallel processors is related to the sizes of time-steps and time-windows, which means more processors will add difficulties to convergence, resulting in degraded speedup, the theoretical speedup limit is bounded to $min\{\frac{m}{I+1}, \frac{p}{I}\}$, which creates a tradeoff between convergence and parallelism [96].

The parallel efficiency E_{pit} of Parareal algorithm is

$$E_{pit}(p) = \frac{S_{pit}}{p} = \frac{1}{\frac{(I+1)p}{m} + I} < \frac{1}{I}, \tag{3.10}$$

which indicates that the maximum parallel efficiency is smaller than 50% due to the fact that $I \geq 2$.

In practice, m must be a large number to compensate overheads caused by coarse-grid and synchronizations. When $(I+1)/ >> I/p$ the theoretical speedup upper limit can be seen as $\frac{p}{I}$, while this limit is still significantly constrained by convergence. An alternative is to use a limited p (10 in this work) and a small time-window ($10ms \times 10 = 0.1s$ in this work).

The PiT+PiS method can retain spatial parallelism in each solution step of the PiT algorithm to improve the maximum parallel efficiency achieved by either method. The speedup of PiT+PiS is given by

$$\begin{aligned} S_{pit+pis}(p_1, p_2) &= \frac{S_{pis}(p_1) \cdot p_2 \cdot m \cdot w_{pis}}{(I+1) \cdot p_2 \cdot w_{pis} + I \cdot m \cdot w_{pis}} \\ &= S_{pis}(p_1) S_{pit}(p_2), \end{aligned} \tag{3.11}$$

where w_{pis} is the execution time with spatial parallel methods, p_1 is number of parallel processors for PiS, and p_2 is the number of parallel processors for PiT. (3.11) indicates that compared to PiT-only or PiS-only method, the PiT+PiS method can utilize more parallel processors to solve a problem with a fixed size. The parallel efficiency for PiT+PiS is expressed by

$$E_{pit+pis}(p_1, p_2) = E_{pis}(p_1) E_{pit}(p_2) < \frac{E_{pis}(p_1)}{I_{pit}(p_2)}. \tag{3.12}$$

where E is the parallel efficiency for each parallel method. In practice, $I(p_2) < I(p_1 p_2)$ is very common, and therefore, $E_{pit+pis}(p_1, p_2) > E_{pit}(p_1 p_2)$. Also, it is possible to achieve $E_{pit+pis}(p_1, p_2) > E_{pis}(p_1 p_2)$ when the PiS thread is saturated at p_1.

Assuming a system with 8 partitions. The parallel efficiency of a PiS method for this system workload is determined by

$$E_{pis}(p) = \begin{cases} \frac{-2}{35} p + \frac{-37}{35} & p <= 8 \\ 0.6 \times 8/p & p > 8 \end{cases} \tag{3.13}$$

which indicates that the system can utilize at most 8 threads for parallel computing.

The PiT method is assumed to have $m = 100, p = 10$, and $I(p) = 2 + int(p/10)$ which means that the iteration number increases by 1 when adding every 10 threads. Substituting these parameters with $p_1 = \{1, 2, ..., 200\}$ into (3.9-3.12) will gives the results shown in Fig. 3.4. Clearly, the PiT+PiS can achieve better performance compared to PiT-only and PiS-only methods. Also, the PiT+PiS method requires more threads so it may be suitable for GPUs with thousands of CUDA® cores.

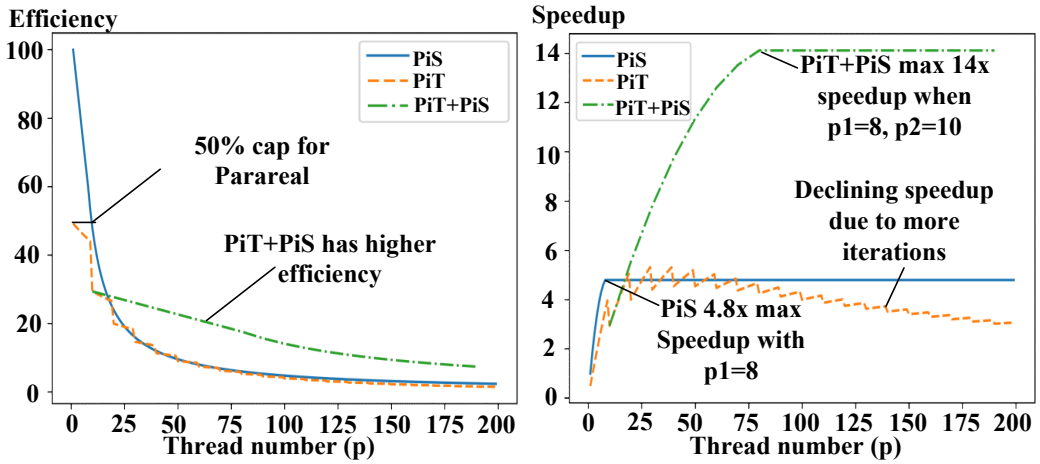

Figure 3.4: Theoretical performance comparisons of PiS, PiT and PiT+PiS methods: (a) theoretical parallel efficiency; (b) theoretical speedup.

3.3 AC/DC Grid Parallel-in-Time-and-Space co-simulation

To establish an AC/DC PiT+PiS co-simulation program, a software hierarchy shown in Fig. 3.5 is proposed. Generally, the AC TS system solver and HVDC EMT system solver are developed independently and connected together via an interface.

3.3.1 GPU PiT+PiS Programming

The thousands of CUDA® cores of an NVIDIA® GPU are affiliated to dozens of streaming multiprocessors (SMs), which are responsible for scheduling instructions, as shown in Fig. 3.6. The frequency of GPU cores is much lower than many prevalent CPUs and hence the instruction cycle is accordingly longer. Considering that the GPU is designed for a large throughput and massively parallel computation, the TS system should be parallelized to achieve a comparable performance to their counterparts on CPU. Therefore, the TS simulation implementation on GPU in this work becomes a unified PiT+PiS scheme.

The pure-GPU computing architecture has been utilized in [23], [24]. However, the pure-GPU Parareal algorithm requires dynamic parallelism and inef-

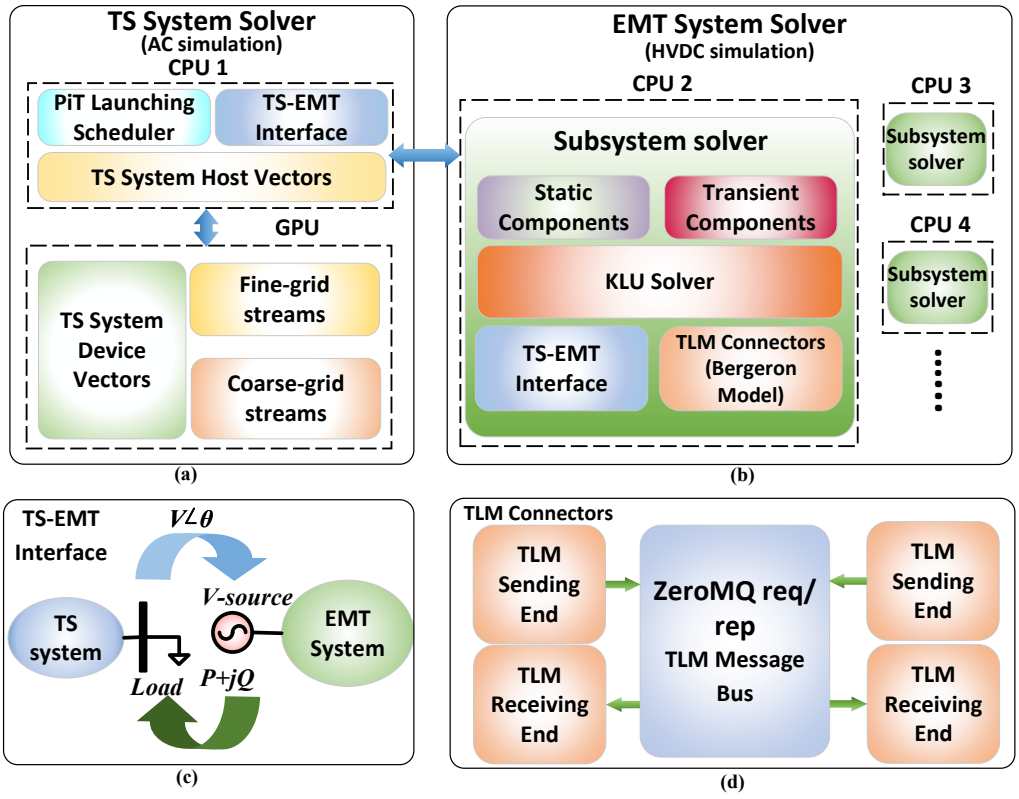

Figure 3.5: TS-EMT PiT+PiS simulation program hierarchy: (a) TS system solver CPU-GPU hybrid structure based on Thrust vectors and concurrent GPU multi-streams; (b) PiS CPU multi-thread EMT system solver structure; (c) TS-EMT interface to connect the TS and EMT solvers; (d) ZeroMQ network to connect TLMs within the EMT system solver.

ficient complex stream synchronizations. For example, the parent coarse-grid under dynamic parallelism will lock GPU resources when launching child grids, and this severely limits the scalability. It is also very slow to perform complex condition checks and loops on GPUs. To avoid degradations, a multi-stream CPU-GPU hybrid PiT+PiS scheme shown in Fig. 3.7 (a) is proposed, which not only utilizes more resources of a single GPU but also enables a multi-GPU architecture. The general pseudocode is attached in Appendix A.

The streams are pre-defined in the initialization stage, and the kernel launching is performed on the CPU. In Fig. 3.7 (a) the coarse-grid stream is labeled as G stream and fine-grid streams are labeled as F streams. For the xth coarse-grid step, its prediction will be used by $(x+1)$th fine-grid kernel function, so that the xth coarse-grid and xth fine-grid kernels can be launched

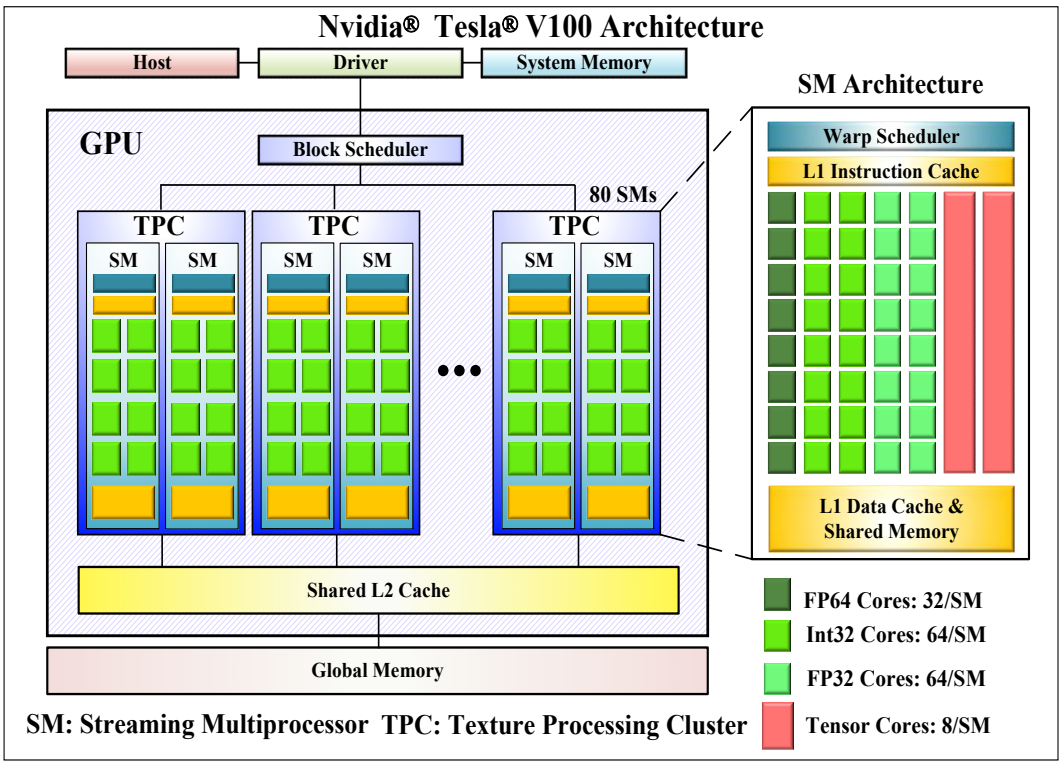

Figure 3.6: General architecture of NVIDIA® Volta GPUs.

at the same time. To maximize concurrency, the refinement of state variables $\boldsymbol{U}$ is integrated into the loop so it can overlap with fine-grid kernel executions. Since $\boldsymbol{U}$ is used in the current iteration to launch the $(x+1)$th fine-grid kernel function, the operation in coarse-grid must be synchronized before $(x+1)$th loop iteration begins, while fine-grid F streams are fully concurrent to each other and G stream. After all coarse operations are finished, the algorithm performs a device-wide synchronization to obtain all fine-grid results which are required in the next iteration. Due to the multi-stream architecture, multi-GPU execution becomes easier since the streams can be assigned to single or multiple GPUs. The memory mitigation problem can be solved by CUDA® unified memory implicitly. The GPU multi-stream execution graph of proposed algorithm implementation is shown in Fig. 3.7 (b).

The memory resources allocation and management are also moved to CPU, which is much easier with the Thrust library [97]. The Thrust library is a GPU-accelerated library of C++ parallel algorithms and data structures; it provides a host-vector class and device vector class as a drop-in replacement for C++

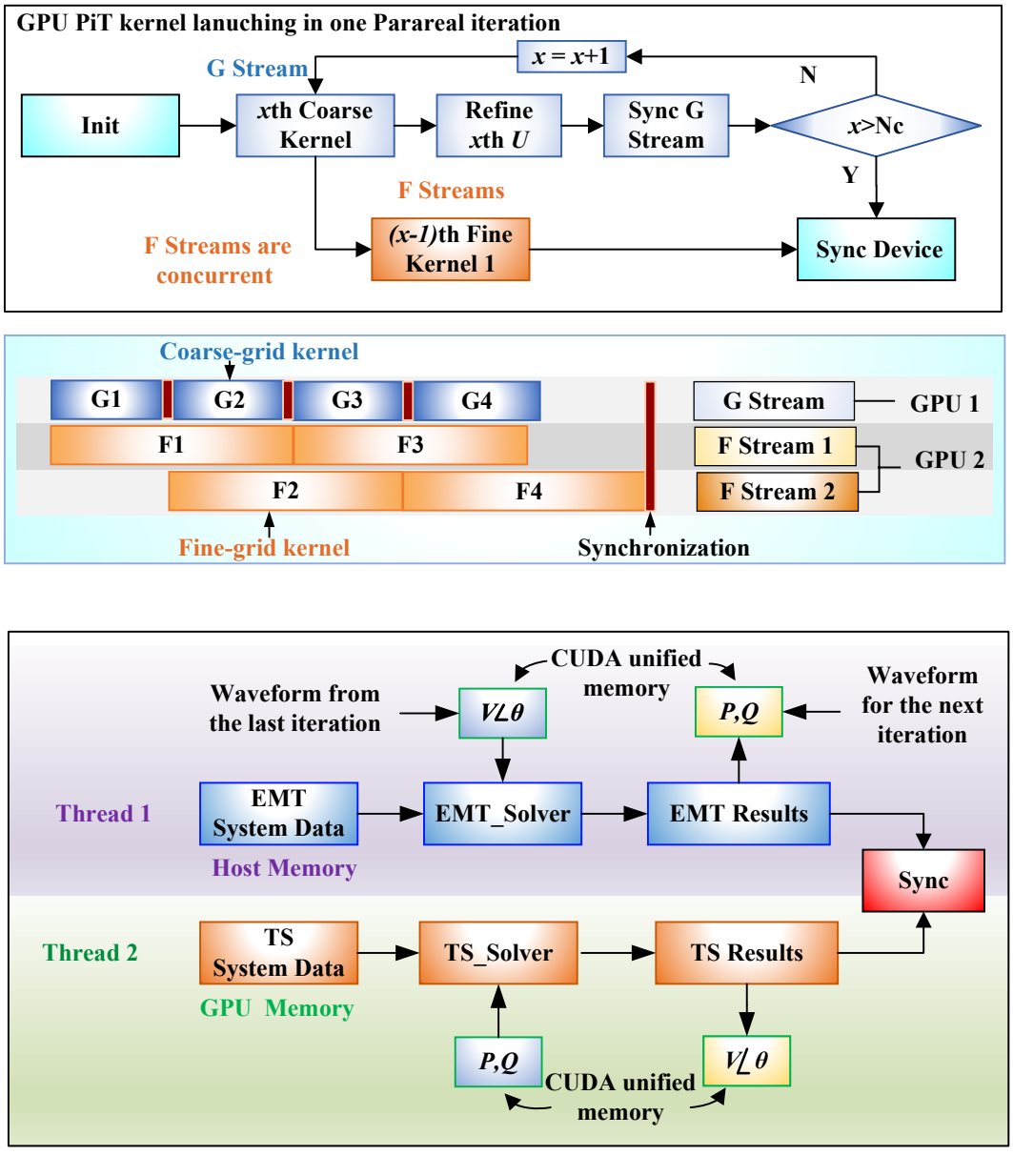

Figure 3.7: Algorithm implementations of the PiT+PiS hybrid CPU-GPU TS-EMT co-simulation: (a) GPU PiT kernel launching within one Parareal iteration; (b) kernel execution graph in multi-stream/multi-GPU scenarios; (c) PiT+PiS TS-EMT co-simulation on hybrid CPU-GPU platform.

std::vector class, which can easily allocate, resize, and transfer memory data between CPUs and NVIDIA® GPUs; it also provides a set of C++ *std* style functions to perform parallel operations on both CPU threads and NVIDIA® CUDA® cores.

In this work, vector classes are used to manage the memory of the PiT+PiS program, and other operations are performed by passing vector device data pointer to the hand-written CUDA® global functions, which execute the simulation step in Fig. 3.3. The host-vector object can be transferred to a GPU device vector object with simple *a = b;* statements; moreover, the vector classes can take an allocator template argument which allocates the vector data memory on unified/managed CUDA® memory, pinned memory, and device global memory without explicit CUDA® function calls. The hierarchy of the TS system solver is shown in Fig. 3.5 (a).

3.3.2 CPU-Based PiS EMT HVDC Simulation

For the EMT solver in Fig. 3.5 (b), static components are time-invariant power system components such as resistors, transient components are time-variant components such as capacitors, inductors, power sources, and MMCs. Due to the asynchronous nature of CPU-GPU hybrid execution and the multi-core parallel computing capability of the CPU, threads running EMT simulation can be concurrent to the TS problem solution on GPU. The granularity of EMT parallel computing is designed to be system-level, which means one thread is responsible for one or more HVDC systems. This can be achieved by OpenMP® task or simple parallel for loop construct. As shown in Fig. 3.5 (d), the decoupling and connections between EMT systems are based on the propagation delay of traveling wave transmission line models (TLMs) such as the Bergeron Line Model. All TLMs have a sending end and a receiving end, which are connected via a message bus implemented by ZeroMQ *req/rep* mode; the sending ends are clients to send data requests to the receiving ends, while the receiving ends are servers to accept requests and replies with their data. The *rep/req* mode of ZeroMQ is synchronous and thread-safe.

3.3.3 AC/DC Co-Simulation Interface

The AC/DC co-simulation method is based on [25] with modifications for PiT purposes, which is shown in Fig. 3.5 (c) and Fig. 3.7 (c). The general idea is that the EMT system is represented as a power source in the TS solution, while the RMS values of the bus voltages in the AC system are transformed into a time-domain instantaneous three-phase voltage source in the EMT simulation. The simulation time-step of EMT simulation is around 50μs while the time-step for TS is 100μs-10ms. Since the TS system only produces voltages and power flows in the frequency domain, the data exchange frequency can be larger than the EMT time-step. However, when using Parareal, the communication between TS and EMT system bring new challenges. The TS PiT simulation consisting of coarse-grid and fine-grid requires the information from the EMT side. If the time-window is small enough such as 1ms or 10ms, the data can be exchanged per window without any iterations. If the time-window is large, it requires restarting the EMT simulation during each typical PiT iteration. The CPU-GPU asynchronous computing architecture enables the EMT simulation to be executed on the CPU while the GPU is solving the large-scale TS problem, so the EMT simulation can be considered as acquired by free: virtually no additional executing time adds to the original TS solving process.

To avoid frequent data copies and synchronizations, the waveforms of interfacing variables are exchanged per-time window; each waveform contains sampled values at the interval of coarse-grid's time-step, which is usually 1-10ms. This method can be considered a combination of Parareal and the waveform relaxation method on AC/DC TS-EMT co-simulation. The two systems exchange the information at each TS Parareal time window as shown in Fig. 3.7. For most steady-state cases, the voltages, and power flow change little so the iteration can converge quickly. For sharp changes such as short circuits faults, the performance only degrades within several time-windows, but the accuracy can be guaranteed since Parareal can fall back to the sequential execution eventually.

The interface data are stored in CUDA® unified memory vectors [98]. The unified memory has a single virtual memory address for CPUs and GPUs. The data mitigation can be triggered by page faults and it is natively supported by the page mitigation engine inside NVIDIA® GPUs later than Maxwell® architecture. The memory can be accessed by CPU and GPUs simultaneously which is suitable for sharing data between multiple GPUs. It also enables asynchronous memory copy with *cudaMemPrefetchAsync* function [99], so that the pinned memory is not required for this task. It saves a lot of complicated memory management works for CPU-GPU and multi-GPU communications and the code can be written as the same as normal multi-thread programs on CPU.

3.4 Dynamic Results and Performance Evaluation

The performance is evaluated based on the test cases shown in Fig. 3.8. The four IEEE 118-Bus systems and a modified CIGRÉ DCS 2 MTDC system form up the Scale x1 base test system, which is used for producing results for study Case 1 and Case 2. Then the AC/DC system is expanded vertically and horizontally as shown in Fig. 3.8 (a) to evaluate the scalability and computational efficiency of the hybrid CPU-GPU PiS+PiT simulation method. The 4 AC power grids in the Scale x1 system are labeled as Net-1, Net-2, Net-3, and Net-4 respectively in Fig. 3.8.

The Scale 2x-12x systems are only used for performance evaluation purposes. Bus 82 is chosen for the connections between HVDC grids and AC systems. For AC-AC connections, the Bus 80 of AC systems are connected. The Scale x1 system contains $4 \times 118 = 472$ TS nodes and 216 generators with four 201-level three-phase EMT modeled MMCs. The TS fine-grid simulation time-step is 100μs and coarse-grid time-step is 10ms; the EMT simulation time-step is 10μs.

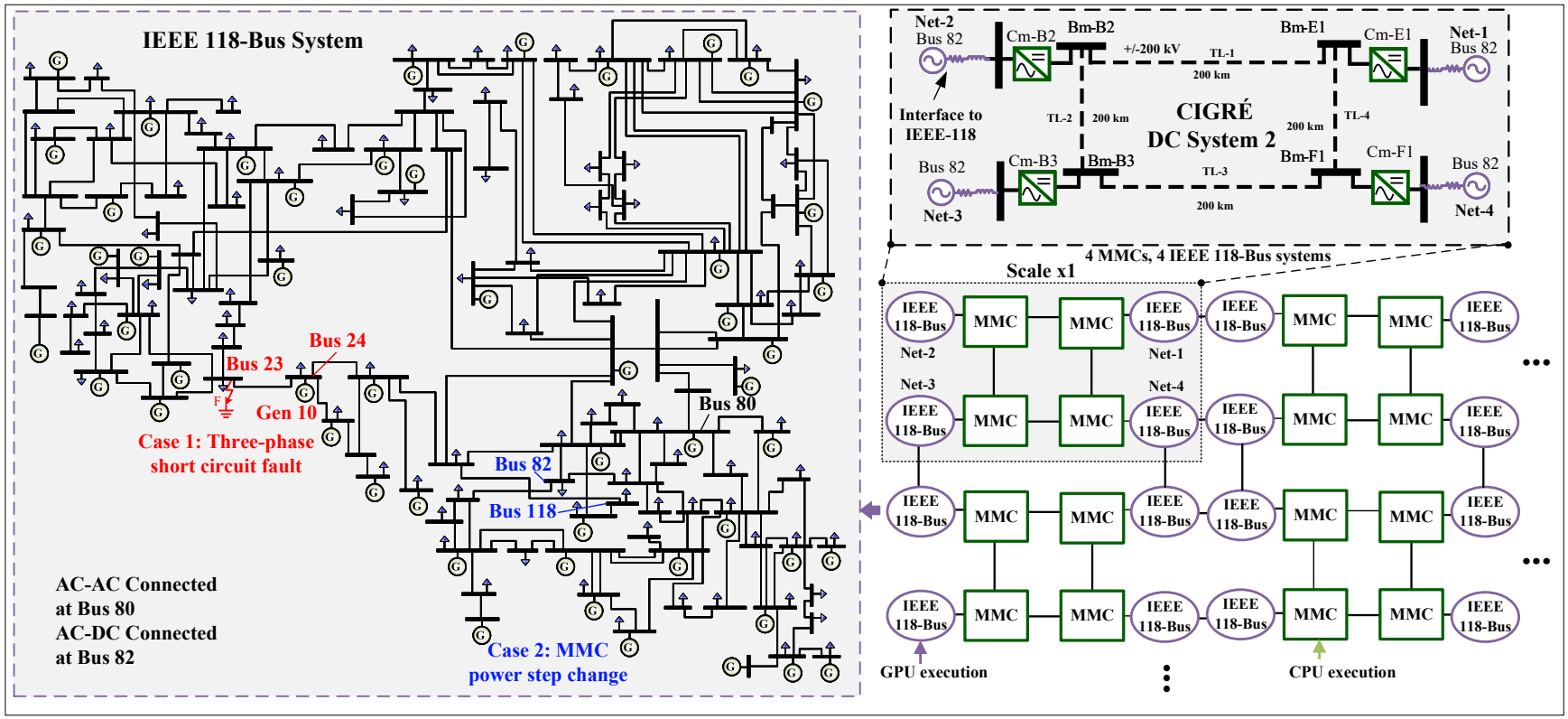

Figure 3.8: Synthetic system for case studies and performance evaluations.

3.4.1 Verifications of PiT and AC/DC Simulation

Case 1: A short circuit with a duration of 200ms happens at Bus 23. The fault resistance is 1Ω; the generator 10 is chosen as a reference for rotor angle. The main focus of this case study is to verify the results compared to the commercial DSATools^{TM} TSAT. The results shown in Fig. 3.9 of the Parareal algorithm meet the TSAT results very well and the relative error is smaller than 1%. After the fault, the generator transients last for two seconds then return to the normal at 10-12s since the fault is cleared. Fig. 3.9 (a)-(c) shows the generator rotor angles, terminal voltages, and frequencies of generators, respectively. Bus 24, which is the closest generator bus to the fault location, has the largest voltage (-0.3 p.u.) and frequency (-0.3Hz) deviations. Fig. 3.9 (d) shows the voltage of non-generator buses, where Bus 23 has the bus lowest voltage but not zero, which is because the fault resistance is 1Ω, not 0.

Case 2: it presents the results of PiT+PiS AC/DC co-simulation of an overload and recovery scenario. The MTDC system is used to support AC power systems when an overload occurs in Net-1; the Cm-B2 is working under the DC voltage control model to maintain constant DC voltages and the other MMCs are working under power control mode. This scenarios has three stages: (1) a 600MW load is added to the Bus 118 at 4s in Net-1, which causes drops in voltages and frequencies as shown in Fig. 3.10 (a), (b) respectively; (2) at 10s the MMC Cm-E1 is ordered to drain 600MW from HVDC buses as shown in Fig. 3.10 (c), thus the voltages and frequencies of Net-1 start to recover. On the other side, Cm-B2 provides the real power of 648MW to maintain DC voltages, and it has to drain real power from Net-2 so that Net-2's voltages and frequencies start to decline as shown in Fig. 3.10 (d), (e) respectively; (3) a 620MW real power is injected at 10.5s to Bus 118 in Net-2 so that Net-2 can maintain the stability.

3.4.2 Performance Comparison

Fig. 3.11 shows the performance comparison between CPU serial AC/DC co-simulation, CPU-GPU PiT+PiS AC/DC co-simulation, and the GPU PiS co-

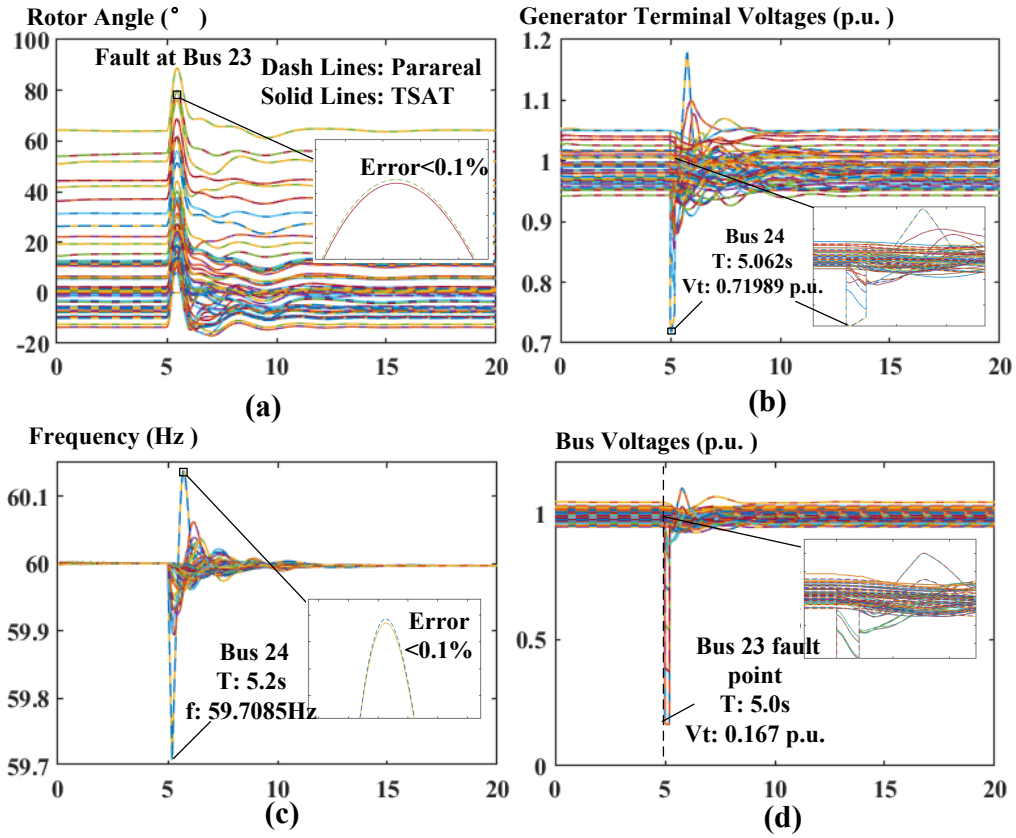

Figure 3.9: Waveform of Case 1 test system: (a) generator rotor angles; (b) terminal voltages of all generators; (c) frequencies of all generators.

simulation with a single NVIDIA® Tesla® V100. The execution time of the CPU serial program for large-scale cases is too long so only the speedup against the serial program is presented in the plot. The speedup and execution time increases almost linearly for both PiT+PiS and the traditional PiS parallel computing method. When the system scale is 12x, GPU PiS only achieved 98.7x speedup compared to the sequential program; meanwhile, the GPU PiT+PiS method achieved 165.6x speedup which is 1.67x faster than GPU PiS.

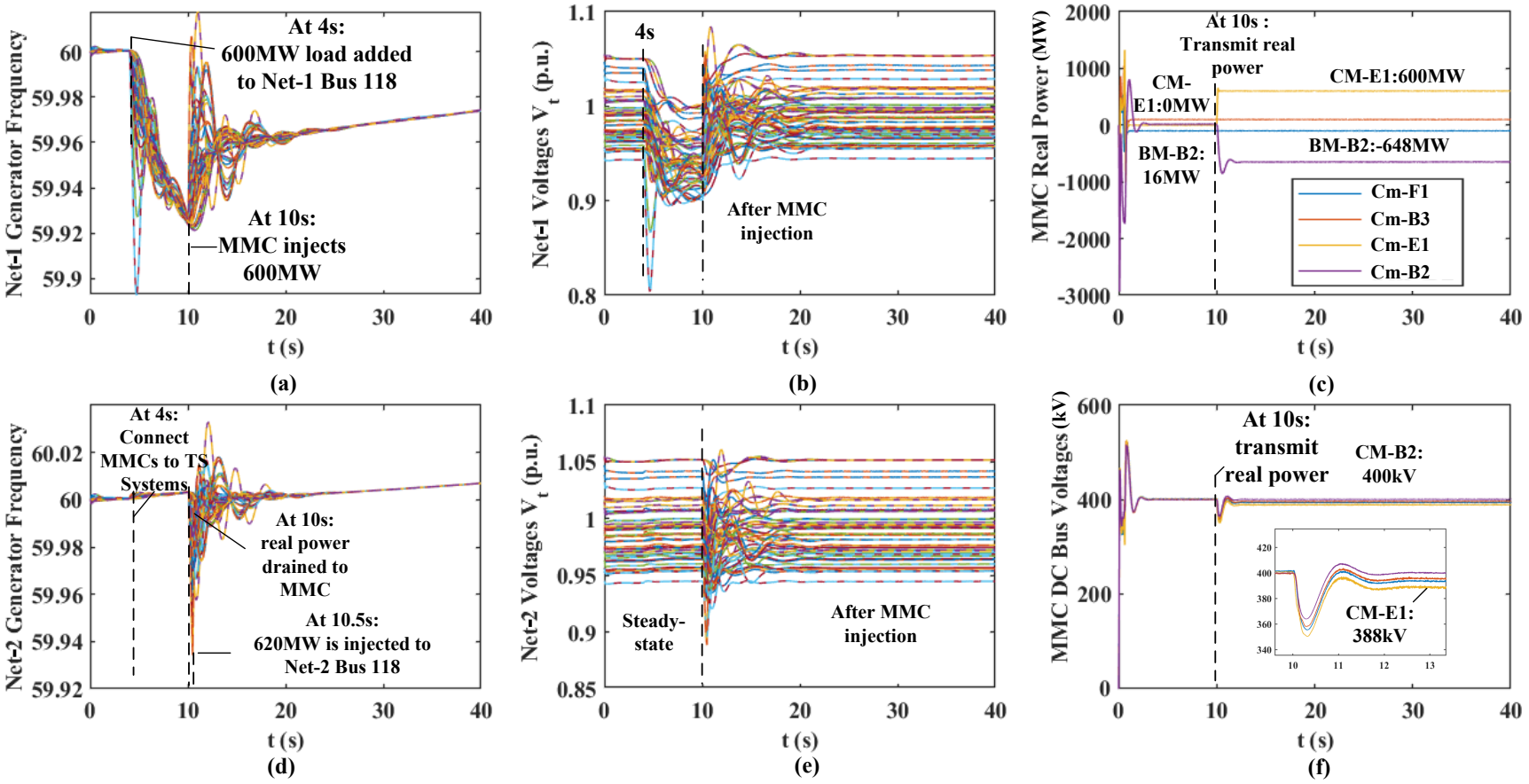

Figure 3.10: Waveforms of Study Case 2: (a) Frequencies of the generators in Net-1; (b) Terminal Voltages of the generators in Net-1; (c) Real power of 4 MMCs; (d) Frequencies of the generators in Net-2; (e) Terminal Voltages of the generators in Net-2; (f) DC Voltages of four MMC terminals.

Fig. 3.12 shows the performance of PiT+PiS method with 2x NVIDIA® Tesla® V100 GPU. When the system scale is 1x and 2x, the speedup is obvious, especially when the system scale is 2x, which has 8 IEEE-118 Bus systems, the parallel-efficiency is near 100%. However, when the system scale is larger, the speedup declines to near 1.0x which indicates the GPU concurrency stops increasing under the multi-GPU situation, despite the single GPU implementation having linear speedup growth. It is due to the size of GPU kernels since the streams in Fig. 3.7 are not guaranteed to be concurrent. The large-size kernel amplified the load imbalance between GPU-1 and GPU-2. Because the serial coarse-grid prediction is critical to performance and it should not be delayed or disrupted, all fine-grid kernels were launched to GPU-2 while GPU-1 only handles coarse-grid tasks. As the problem size grows, coarse-grid workloads become much smaller than fine-grid workloads, so the multi-GPU results become closer to single GPU execution. The more advanced solution is to launch some of the fine-grid kernels to GPU-1 when it is idle so that the computationally intensive fine-grid tasks can make use of multiple GPUs.

3.5 Summary

A hybrid CPU-GPU parallel-in-time-and-space transient stability simulation method is proposed based on the Parareal algorithm. The Parareal algorithm is implemented on GPU along with the traditional PiS algorithm to achieve maximum parallelism. The CPU-GPU design performs PiT scheduling and launches GPU kernel functions to streams on the CPU, which brings better scalability and extensibility to GPU-only design. Meanwhile, the CPU can perform the EMT simulation asynchronously when the GPU is running the transient stability simulation, which can be perfectly integrated with the proposed AC/DC co-simulation scheme and bring better performance and parallel efficiency. The study case shows good results both in accuracy and computational performance. The speedup for the PiT+PiS method to the PiS method is around 2x and can achieve 165.6x compared to sequential CPU programs for a large-scale system. The method can utilize multiple GPUs and can achieve

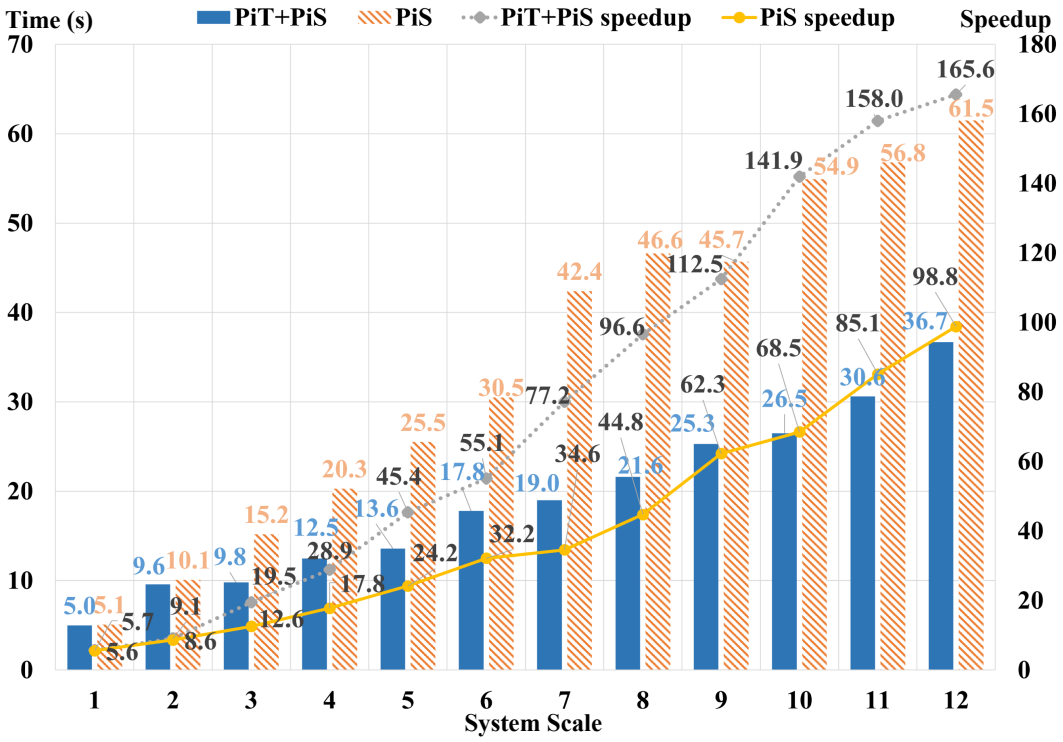

Figure 3.11: Performance comparison of hybrid PiT+PiS and GPU PiS under various system scales.

near-maximum parallel efficiency with a system scale of 2x. Further investigations and benchmarks are planned to find the bottleneck and optimize the PiT+PiS algorithm on multi-GPUs. The proposed hybrid PiT+PiS method shows good potential for the solution of large-scale AC/DC power system transient stability simulation problems.

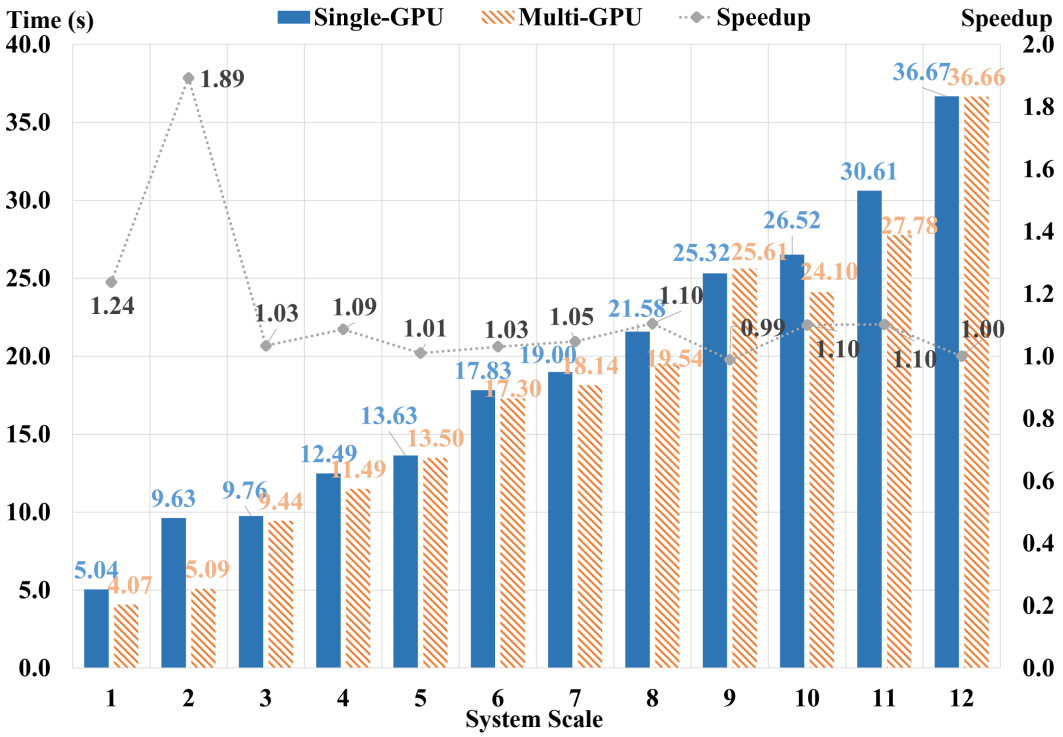

Figure 3.12: Performance of the multi-GPU implementations.

Chapter 4

ECS-Grid: Data-Oriented Real-Time Simulation Platform for Cyber-Physical Power Systems

4.1 Introduction

[1] The chapter is organized as the following: Section 4.2 introduces the fundamental architecture and methodologies in ECS-Grid. Section 4.3 introduces the simulator MessagePack protocol and the performance test of vIEDs with different transportation protocols; the way to implement industrial protocols such as IEC-60870-5-104 is also discussed. Section 4.4 presented the microgrid cluster study case with results from one steady-state scenario and one cyber-attack scenario.

The transition to clean and renewable energy in the power industry is playing a significant role in reducing the emission of greenhouse gases and fighting climate change [100]. However, the traditional power systems that rely on centralized control networks and controllable generators become insufficient to meet the challenges of future power systems such as microgrids with the high penetration of uncertain and unstable renewable energy [101]. Therefore, new intelligent decentralized control solutions based on modern information and communication technologies are emerging to face the new challenges [35],

[1] This work has been published: **T. Cheng**, T. Duan and V. Dinavahi, "ECS-Grid: Data-Oriented Real-Time Simulation Platform for Cyber-Physical Power Systems," *IEEE Trans. on Ind. Informat.*, vol. 19, no. 11, pp. 11128-11138, Nov. 2023, doi: 10.1109/TII.2023.3244329.

[102], [103]. The new research works heavily involve communication between control centers and IEDs, which are the foundation of smart grid and power system automation [42]. Thus, detailed and accurate real-time simulation of CPPSs [104] is necessary for future power system research[41], [105], [106]. However, the scalability, flexibility, and performance of the traditional power system analysis tools and communication analysis tools are inadequate. For example, the existing software-based simulation approaches such as EPOCHS [107], GECO [37], INSPIRE [36] and the simulators proposed in [38], [108], [109] aim to create a network interface for existing power system simulation tools and glue the two domains in power grid simulator and communication network simulators (NS-2, NS-3 and OPNET et al.), which cannot reflect the behaviors of IEDs in real-time environments and must handle the complicated synchronization between two different simulation domains. Some works such as [39], [110]–[112] bridged the commercial real-time power system EMT digital simulators to the communication simulation systems, which can achieve real-time performance and more realistic behaviors. However, the high-cost commercial EMT simulators were designed for industrial verification purposes and still lack the scalability and flexibility for CPPS-related academic research.

The essential problem is that both power grid and communication network simulation tools are initially designed for their single domain. The traditional software mainly based on OOP has become a huge obstruction to building a native and comprehensive cyber-physical simulation platform. The industrial programs are dealing with various forms of data and their combinations, while the OOP paradigm emphasizes predetermined inner structure and relationships of objects. Plain data such as an array of float numbers can represent many things in the computer world. On the opposite, a class and its object can only be used for one purpose predefined by abstract templates, which brings significant difficulties to repurposing existing designs and thus cannot elegantly describe and solve the problems in the complex interdisciplinary CPPS.

Therefore, this chapter presents the ECS-Grid: a novel real-time cyber-physical EMT simulation platform with virtual IEDs (vIEDs) based on the cutting-edge entity-component-system (ECS) software framework. The pro-

posed ECS-Grid simulation platform has the following major advantages:

1. High Flexibility: Compared to the traditional dominating object-oriented paradigm which is based on *Polymorphism, Abstraction, Inheritance, and Encapsulation*, the ECS framework is based on a data-oriented paradigm: *Entity* (usually an integer), *Component* (pure data structure), and *System* (plain functions to perform algorithms on components), where entities are defined by the combination of data components, and component functionalities are defined by systems. This data-oriented paradigm avoids dependency complexities caused by OOP inheritance and brings flexible model description ability. Any data component and the system can be replaced not only at the compiled time but also at the run time. Such a feature is highly desired for cyber-physical simulation since the various form of data and data flows are the major concerns. For example, real-world IEDs are composed by multi-functional circuit boards and these replacable boards can be represented by data components on a vIED entity.

2. High Extensibility: with the advanced data-oriented design, components and systems are grouped into plugins in ECS-Grid, and a simulation application is composed of a set of plugins. In contrast to traditional software which often provides a huge library as an undividable whole, ECS-Grid allows users to only pay for what they need. Although it is initially designed for CPPS simulation, it can run pure physical simulation similar to simulators without cyber layers, or run the cyber features for other purposes without the physical EMT simulation. Moreover, the users can create plugins easily even in dynamic libraries with their customized components and systems, and add or override core functionalities such as the matrix solvers or additional communication protocols.

3. High Scalability: With the benefits from the ECS framework, a vIED layer is proposed which mainly utilizes scalability protocols from the message-oriented asynchronous ZeroMQ [113]. A MessagePack-based [114]

JSON-like simulator protocol is proposed for the simulator to bridge various industrial protocols. The utilization of the middleware makes it easy to scale ECS-Grid from a single CPU node to multi-thread applications or even distributed networks which resembles real-world automation systems. The performance test on a single-thread ZeroMQ vIED with MessagePack-based protocol shows a minimal latency of $6\mu s$ and an average latency of 20μs with an effective 60Mbit/s bandwidth, which is quite enough for a wide range of application scenarios.

A 711-node AC/DC microgrid cluster based on the modified CIGRE-15 Bus microgrid system with a man-in-the-middle cyber-attack scenario is set up for demonstration and performance evaluation. The simulation results show that the proposed solution can achieve faster-than-real-time (FTRT) performance on 10th Gen Intel® Core™ i7 CPUs and real-time performance on NVIDIA Jetsons with dual-core ARM v8 CPUs.

4.2 Proposed Data-Oriented Architecture of ECS-Grid

4.2.1 Data-Oriented ECS Architecture

The information exchange between physical and cyber systems is the major concern in a cyber-physical simulation. Information is carried by data, and generally, cyber systems are built to transport and process data that carry useful information. However, while data can represent almost anything in the digital world, an object which contains both data structure and behaviors can only carry limited information whose pattern is pre-defined by its abstract templates: the *Class*, without the ability to mutate its structure. The *Inheritance* makes it even worse due to the extra dependencies between *Classes*. As shown in Fig. 4.1 (a), the OOP paradigm creates abstract base *Classes* for different domains while inherited implementations are realized in sub *Classes*. This adds difficulties in refactoring and optimization. Since cyber-physical systems in the big data age are transporting enormous unstructured data, a

data-oriented solution that focuses on data processing and data combinations is highly preferred to OOP solutions.

Data-oriented programming means data combinations determine functionalities, which is also the core concept of ECS-Grid. The ECS framework starts to play a significant role in the game industry and modern software engineering, which is now the backbone of Minecraft™ [43], Data-Oriented Technology Stack (DOTS) in Unity® [44], Call of Duty®: Vanguard, ArcGIS Runtime SDKs by Esri™, and many modern large-scale commercial software projects. However, it is still not utilized for cyber-physical simulation in power industries which are full of data-intensive applications. Currently, there are three major types of ECS frameworks: bitset, archetype, and sparse-set, where sparse-set is the most popular one due to its high flexibility and archetype has the best theoretical performance. The specific types of components are managed by an entity registry to provide database-like access to the data objects. In this work, the sparse-set-based EnTT [43] is used as the entity registry, which is also used in Minecraft™. Everything under the ECS framework belongs to an *Entity*, *Component*, or *System*. The inheritance is replaced by an entity's composition of data components in the ECS framework. An *Entity* is an integer identifier that is linked to multiple components in an entity data registry which can be seen as a data table in Fig. 4.1 (b). A *Component* is a structure with data to process. As shown in Fig. 4.1 (b), entities are rows of the data table. For example, a voltage-source converter (VSC) entity is composed of the four circles in a row of the table view; the EMT model object which is the same circuit object of the traditional OOP design such as an averaged-value model VSC; the IO module which holds measured signals and controller signals; the VSC controller holds data for control logics and the IED communication module holds the information of network sockets and other parameters such as latencies.

A specific combination of data components will be processed by relevant *Systems* which contain all program logic. A *System* is a plain function that can process columns of components in the table of Fig. 4.1 (b). It usually takes the registry as the input parameter and creates a query view of compo-

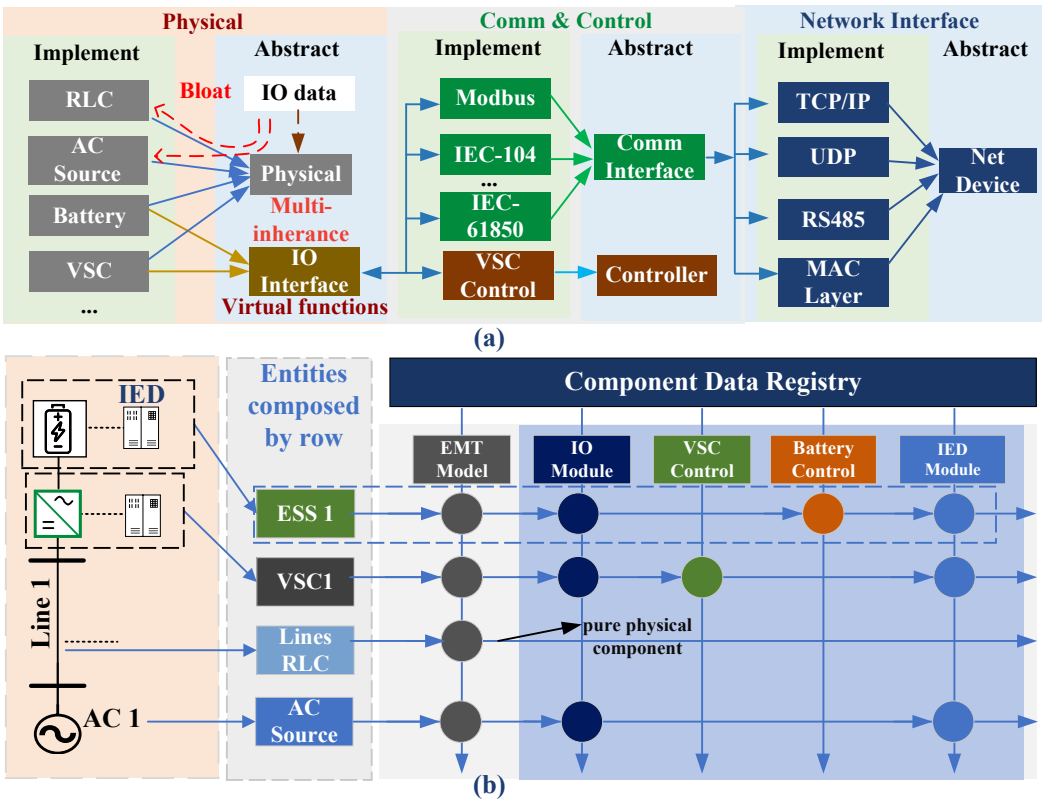

Figure 4.1: Traditional OOP and data-oriented ECS design for CPPS simulation: (a) Inheritance and abstract interfaces create complex object relationships to represent physical objects with IED under the OOP paradigm. (b) An entity is defined by data component combinations similar to a row in a data table under the ECS framework, while data are processed by columns.

nents from the registry just like a database query. The queries are optimized by the ECS framework depending on its storage type and usually are blazing fast. Because the system function is only called once on specific types of components, it eliminates the bloating issue caused by intermediate interfaces or CPU overhead in dynamic virtual function calls per object; since the components are stored in compact arrays, it is also more cache-friendly and easier to take advantages of modern single-instruction-multiple-data (SIMD) hardware such as graphical processing units (GPUs). In other words, it can fully avoid the usage of inheritance and polymorphism to build more complex and efficient software with an ECS framework. Also, ECS brings impressive flexibility to modify an *Entity*. For example, one can replace the VSC control component without breaking other VSCs with the same physical model, or

replace a system at the runtime to change the functionality.

The proposed ECS-Grid currently uses a hybrid ECS solution to fully reuse the traditional OOP EMT simulation code. The EMT simulation loop is untouched, and no modification is added to any physical component class. The only difference is the traditional physical components are now managed as a part of an entity in an ECS registry instead of an all-in-one object. The IED feature is added by introducing new components and systems to the physical software. This hybrid solution can be very useful for industrial developers to transfer from traditional OOP to data-oriented design under the ECS framework. The full transition to an ECS data-oriented simulation framework requires many critical changes to traditional design patterns and still needs some explorations.

4.2.2 Data-Oriented IED Simulation

In most CPPS simulations, there were only two layers: the physical layer and the cyber layer, which often ignored first-class citizens in real-world CPPSs: intelligent electronic devices (IEDs). The IEDs are generally protection, control, and monitoring devices in power grids, which are cornerstones for the modern power system automation and smart grid [115]. Therefore a comprehensive cyber-physical simulation should model these IEDs to be as realistic as possible.

As shown in Fig. 4.2, a commonly used IED in power systems is composed of multiple modules: *DSP controller*, *CPU*, *Network DSP (only for optical IEC-61850 GOOSE/SV)*, *AI, AO, DI, DO* for analog or digital inputs/outputs (IOs), and the *power source*, which are circuit boards is responsible for specific tasks. Different configurations of the board modules and internal firmware will define the functionalities of the IED. It can be beneficial to model the IEDs inside EMT simulation to reflect real-world communication behaviors such as network latencies, time synchronizations, and protocol analysis, and also bring many new possibilities to cyber-physical research. The structure of real-world IEDs is a perfect match for the proposed data-oriented architecture.

ECS-Grid proposed the layer of vIEDs, which is an independent set of

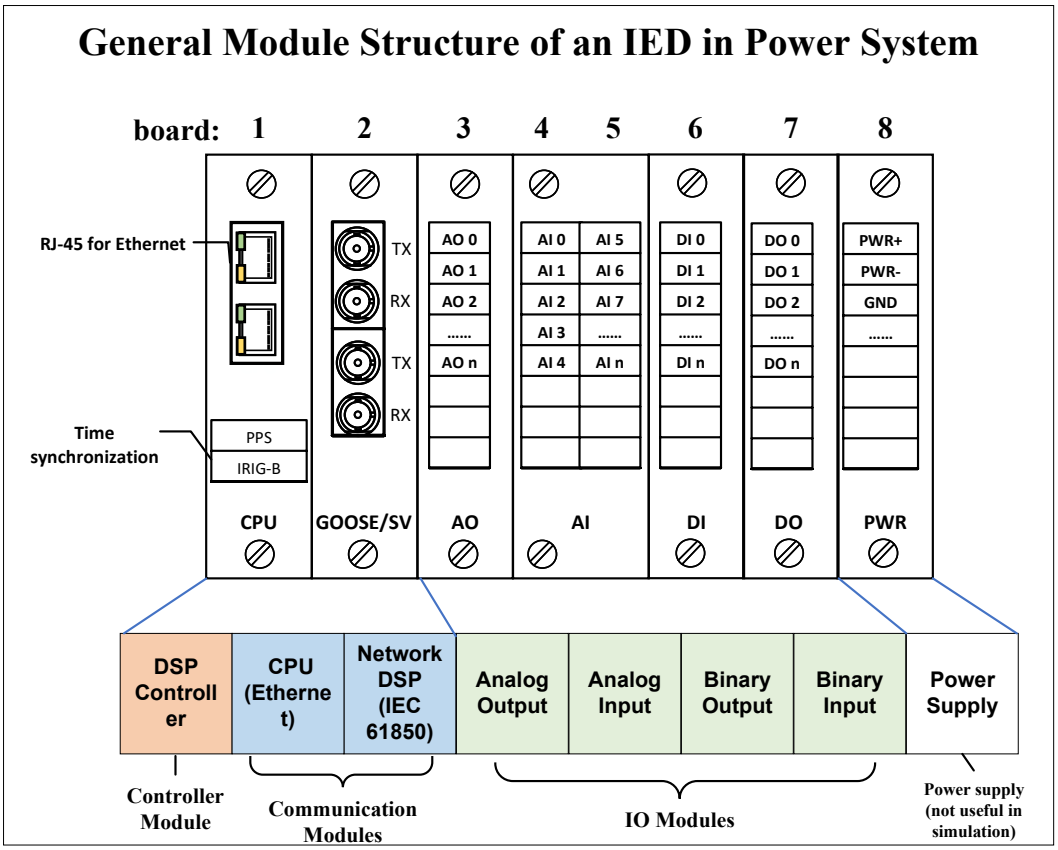

Figure 4.2: A real-world IED consists of the controller, communications, IO modules, and power supply.

components and systems to model IEDs in power grids to conduct control or communication tasks. These digital twins of real-world IEDs bring more realistic and more consistent experiences from real-world CPPS. The vIEDs consist of IO, control, and communication components in the ECS-Grid. which covers the fundamentals of a real-world IED in Fig. 4.2.

As shown in Fig. 4.3, the components of VSC entities in Fig. 4.3 are similar to the modular boards in Fig. 4.2; The systems: *VSC_IO*, *VSC_Control* and *VSC_IED* are similar to the software programmed into the physical IED; the systems are grouped and called in the *IED Stage* which is considered as a new sensing layer compared to old physical simulation loop. Similar to the replaceable boards and upgradable programs in real-world IEDs, the components and systems of vIEDs can be replaced or reorganized to serve different purposes both at compile-time and runtime. This is realized elegantly

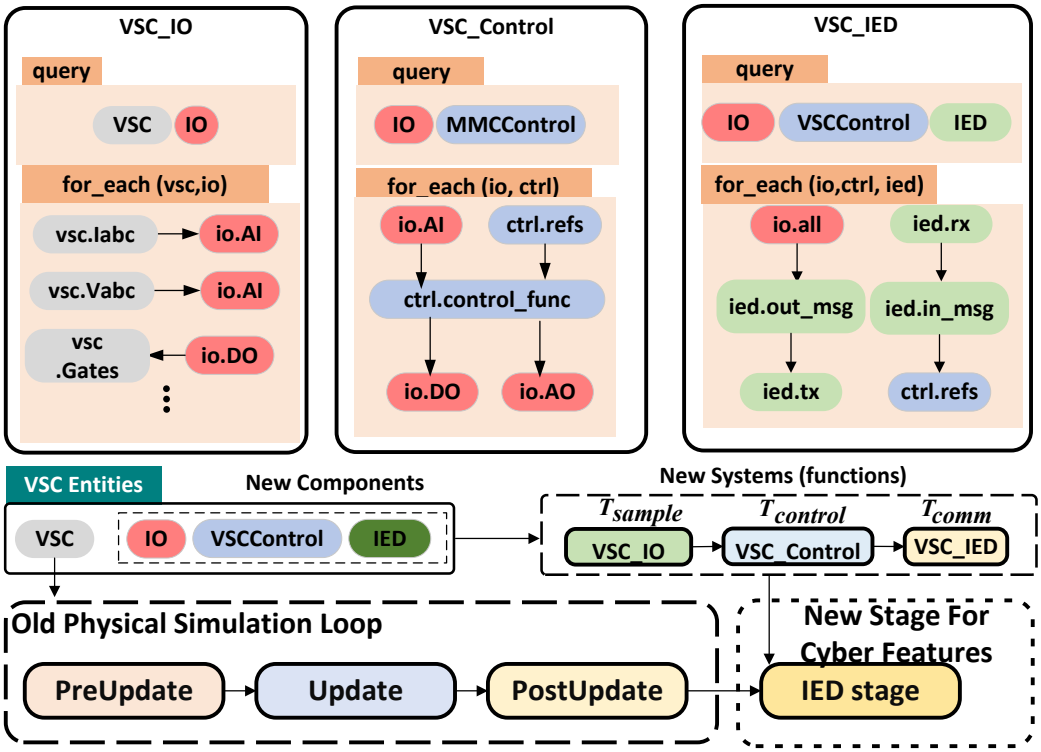

Figure 4.3: IED sampling, control, and communication systems execution in the proposed data-oriented framework of ECS-Grid.

within a data-oriented ECS framework while OOP cannot compose it nicely due to its fixed pre-defined structures. The vIED extension is implemented by a very simple plugin interface introduced in Section 4.2.3.

4.2.3 Plugins Made Easy

The extensibility is important for a cyber-physical simulation platform and that should be a significant advantage of a data-oriented design. The functionalities of ECS-Grid are defined by a combination of plugins, which is similar to many popular ECS frameworks such as Flecs and Bevy. Plugins can have various inner structures and definitions as long as they provide a plain function with a declaration of *void build(World &world);* as the entry point. In this way, a plugin with functionalities in Fig. 4.3 can be loaded from a header-only library, a static library, and even a dynamic library loaded at run-time. The implementations are quite straightforward and an example C++ header of the ZeroMQ vIED plugin is included in Appendix B .

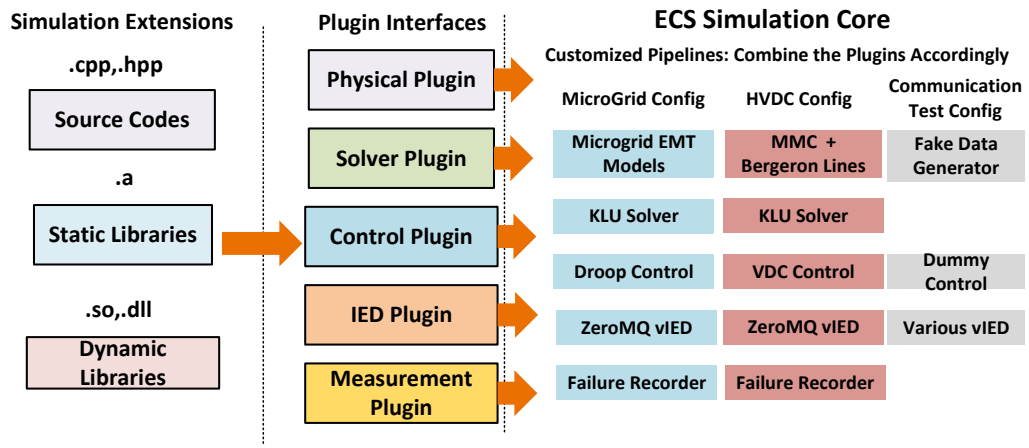

Figure 4.4: Example plugins and their configurations in the proposed ECS-Grid.

As shown in Fig. 4.4, a simulation based on the ECS framework is composed of various plugins, which is flexible and bloat-free. For example, although the solver and vIED plugins are the same, the simulation for microgrids uses exclusive microgrid systems such as renewable energy sources and storage units along with the droop controllers, while the HVDC simulation configuration only uses the MMC and Bergeron line model plugins. To test the vIED only, the physical plugins are removed and replaced by dummy data sources. These configurations are practically applied to produce the results presented in Section 4.3 and 4.4.

4.3 Proposed Data-Oriented Protocol for Real-Time Cyber-Physical Simulation

Traditional CPPS simulations are usually based on available commercial simulators, where the signals of the physical power grid are grouped, converted to industrial protocols, and sent to cyber simulation machines. However, the industrial protocols are designed for production environments which should consider security issues, standard requirements, guidelines of power system operations, and the limits of existing industrial communication routes and devices. However, the simulation environments should provide a more generic protocol to simulate various scenarios which cannot be covered by a single

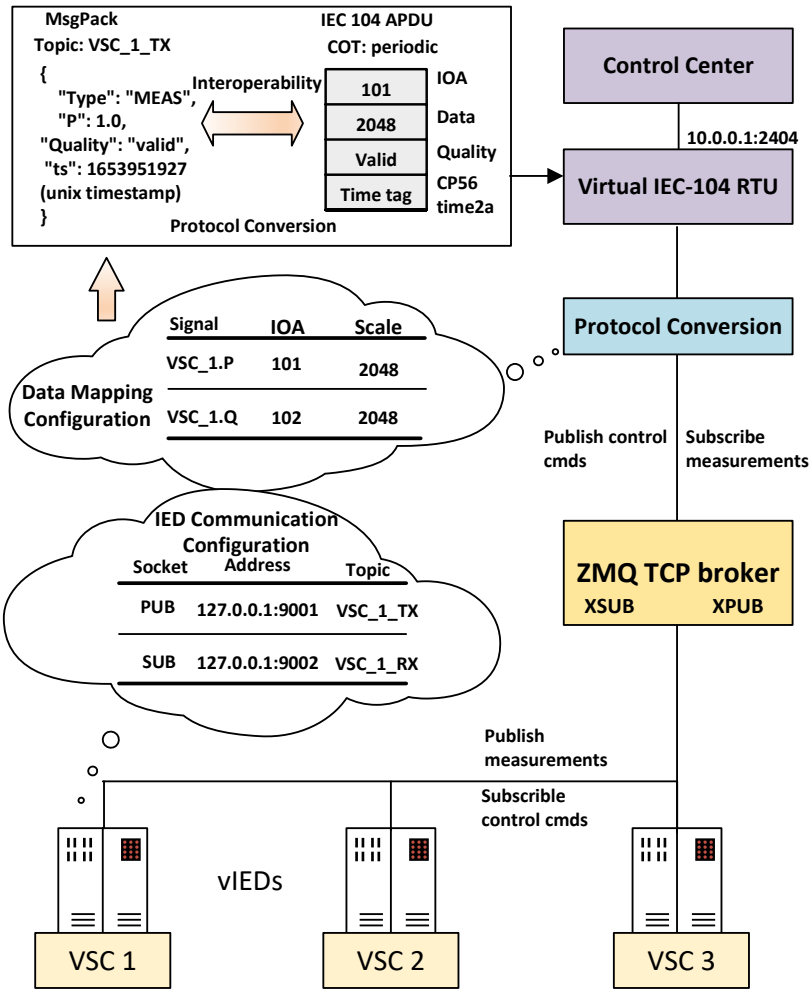

Figure 4.5: The communication between vIEDs and real-world IEC 60870-5-104 clients.

industrial protocol. Also, the simulator itself should provide exclusive remote control and management functionalities for simulation-only purposes which are not considered by industrial protocols and IEDs.

Although some platforms [108] use Open Platform Communication (OPC) or CORBA (Common Object Request Broker Architecture) DIM (Distributed Information Management) protocol to unify the protocols within the simulator, these traditional OOP protocols are based on late-1990s standards and technologies which cannot meet the data-oriented demands of modern cyber-physical simulation. The OOP protocols often need a cumbersome object library to decode the messages and many functionalities are fixed. Therefore, a data-oriented protocol and a local simulation network are proposed

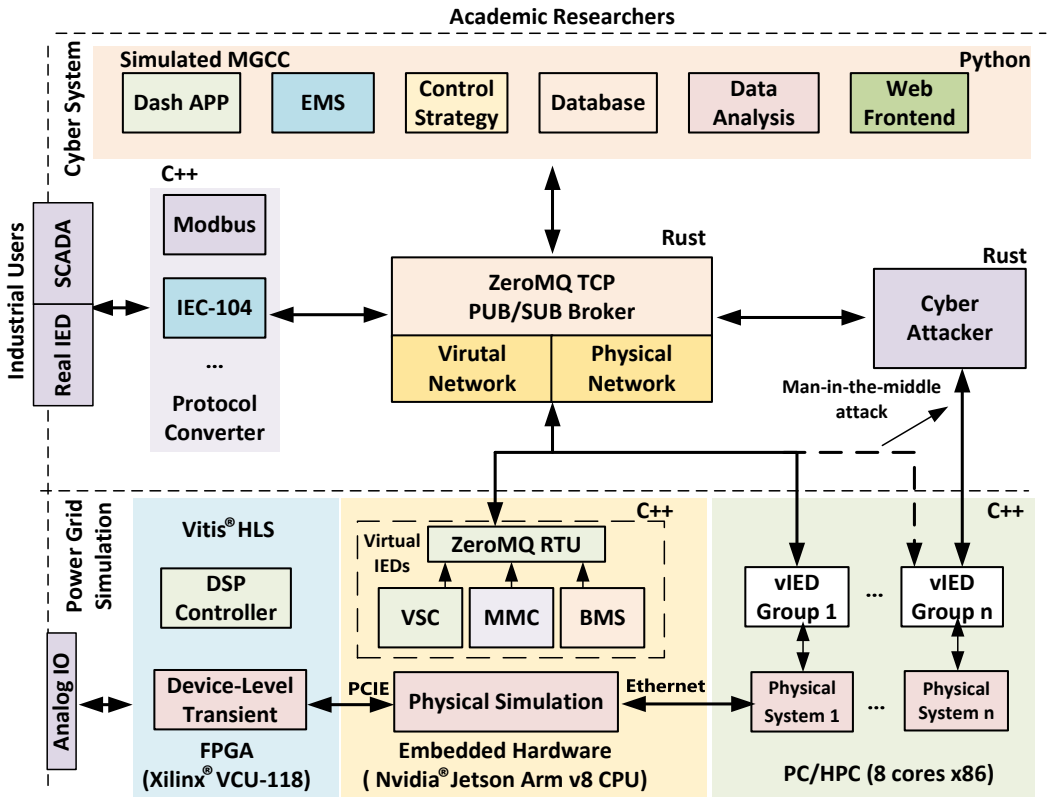

Figure 4.6: The distributed network architecture of the ECS-Grid platform.

for the vIEDs as a unified middleware interface to the outer systems. The data-oriented protocol should be:

(1) generic: it should be able to represent different messaging patterns used in microgrids and not be restricted to specific transport media;

(2) high-performance: it should not add heavy overhead to the simulation systems and can handle a large volume of data; it should have distributed and concurrent features to make full use of modern hardware;

(3) customizable: unlike industrial protocols where all are defined by standards, the CPPS simulation should enable more possibilities for research explorations of future power systems by allowing users to customize the protocol.

4.3.1 MessagePack Format for User Applications

The ZeroMQ mainly abstracts the sockets for higher-level applications, the payloads being transported depends on the user's decision. The default vIED plugin uses a simple solution based on javaScript object notation (JSON)

and MessagePack is proposed for a generic and customizable application-layer simulation protocol. JSON is the first-class data format inside ECS-Grid for configurations and data exchanging shown in Fig. 4.5. It is faster and smaller than the current XML format used in industrial applications[116]. The JSON format is self-describing, so there are no complex data models predefined by an Interface Definition Language (IDL) to decode the messages. MessagePack is an efficient binary serialization format and it can exchange JSON data cross multiple languages more efficiently [114]. The receivers can easily decode the messages to JSON objects like dictionaries in Python, ECMAScript, and Rust and handle them in their program logic. The utilization of MessagePack can provide a faster serialization and deserialization speed without a pre-defined schema and reduce the size by more than 40% compared to a plain-text JSON message. The example for the JSON-like protocol and the protocol conversion. The MessagePack design enables users to simulate specific scenarios and make custom virtual cyber services such as microgrid control center (MGCC) upon vIEDs, which can provide a very convenient platform for developing future distributed multi-layer control schemes and other cyber components as shown in Fig. 4.6.

Real-time communication performance is easy to achieve since simulation environments have much better computing power, bandwidth, and reliability than field devices. Modern CPUs have quite a large memory bandwidth that is larger than high-end optical networks. For example, Intel® Core™ X-Series Processors can achieve a bandwidth of 94GB/s with 4-channel DDR4 2933Mhz memories [117], which is nearly 8 times faster than high-end 100Gbit/s Ethernet. The current 10/100Mbps industrial Ethernet bandwidth is no match to the CPU's internal bandwidth.

Although the customizable protocols are useful for simulation environments, the industrial protocols cannot be ignored. As shown in Fig. 4.5, protocol converters are the solutions, which map the MessagePack protocol to a specific protocol such as IEC 60870-5-104. Protocol converters are common in real-world power automation systems and many IEDs can do protocol conversions internally according to the firmware or hardware configurations.

Thanks to the multi-transportation ZeroMQ, the protocol conversion can have multiple choices to meet users' demands and interoperability can be ensured by customizing MessagePack messages. If users want industrial protocols directly built into the IED, they can follow the same plugin development principles to integrate their protocols.

4.3.2 Comparsion of Various Middleware Protocols

Currently, there are three communication plugins available for vIEDs in ECS-Grid: ZeroMQ [113], eProsima Fast DDS Real-Time Publish-Subscribe protocol (RTPS) [118], and Eclipse Paho MQTT [119]. Fast DDS is the middleware used in Robot Operation System 2 (ROS2). The MQTT is used for Internet-of-Thing (IoT) applications and partly in microgrid applications with IoT devices. ZeroMQ is a widely used message-oriented middleware. The latency test results of different protocols under the one-publisher-one-subscriber vIED scenario are listed in Table 4.1-4.5.

Table 4.1 shows the results from the Fast DDS RTPS protocol. There is a spike in maximum latency when the message number increases, which is normally due to unreliable User Datagram Protocol (UDP) transportation. In summary, RTPS's performance is high and stable, and it has advanced features which can be very useful for vIED applications. However, it requires many dependencies, and the provided advanced features are not used in power systems. Moreover, it is not easy to use and the support documents should be greatly improved compared to other solutions.

Table 4.2 shows the results from Eclipse MQTT Paho clients. The MQTT is not designed for microsecond-level latency, and it must have a broker, which is an Eclipse Mosquitto broker[120]. The default configuration also enables message persistence on the broker server. Therefore, the latency is 1-100ms level which is good for most IoT applications but not good for low-latency communications. However, the bandwidth reaches the top of all protocols when the published message number is 1000. In all, the MQTT solution can be useful for some IoT scenarios since not all devices need microsecond-level latency.

Table 4.1: vIED Latency and Bandwidth Using eProsima Fast DDS (RTPS)

Messages	Max (μs)	Min (μs)	Mean (μs)	Pub Bandwidth (Mbit/s)
100	98.70	14.42	16.99	34.28
1000	211.44	12.77	15.80	36.96
10000	293.81	13.11	14.77	38.15
1000000	7483.24	11.88	15.06	38.63

Table 4.2: vIED Latency and Bandwidth Using Eclipse Paho MQTT

Messages	Max (ms)	Min (ms)	Mean (ms)	Pub Bandwidth (Mbit/s)
100	99.81	95.75	96.86	22.21
1000	99.87	75.60	92.67	71.41
10000	140.22	27.90	80.04	18.34
1000000	145.71	0.92	78.44	15.20

Table 4.3- 4.5 shows the ZeroMQ vIED performance under different configurations. The in-memory inter-thread communication reaches the lowest latency of 6μs, which is quite enough for IEC-104 applications since the protocol time-stamp has a resolution of milliseconds. The ZeroMQ point-to-point TCP pub/sub latency is around 20μs and the TCP pub/sub with a broker test is just a doubled point-to-point TCP latency. The single-thread publisher's bandwidth is high and stable without throughput optimization and well suited for a real-world IED which mainly has a 100Mbit/s Ethernet port. Besides, ZeroMQ is quite flexible and easy to use in every major programming language. The only problem is it requires more user decisions to establish an in-production network, however, it is an advantage for a simulator that can give users the maximum freedom to establish customized scenarios.

For the generic and high-performance goals, ZeroMQ is recommended to be the message bus between vIEDs. ZeroMQ is a high-performance asynchronous messaging library, aimed at use in distributed or concurrent applications. ZeroMQ supports scalability protocols (pub/sub, request/reply, client/server, and others) over a variety of transports (TCP, in-process, inter-process, multicast, WebSocket, and more). This keeps the code clear, modular, and scalabe from very low-latency in-memory communication to the large-scale cloud com-

Table 4.3: vIED Point-to-Point Latency and Bandwidth Using ZeroMQ (Inter-Thread)

Messages	Max (μs)	Min (μs)	Mean (μs)	Pub Bandwidth (Mbit/s)
100	203.21	6.63	25.23	18.28
1000	210.52	6.69	10.26	47.91
10000	191.43	6.37	7.88	60.40
1000000	230.29	6.05	7.08	67.03

Table 4.4: vIED Point-to-Point Latency and Bandwidth Using ZeroMQ (TCP)

Messages	Max (μs)	Min (μs)	Mean (μs)	Bandwidth (Mbit/s)
100	366.13	20.76	84.35	17.50
1000	396.42	16.44	24.87	48.28
10000	347.94	15.49	20.70	60.59
1000000	332.94	15.04	18.48	62.83

Table 4.5: vIED Latency and Bandwidth Using ZeroMQ (TCP with Broker)

Messages	Max (μs)	Min (μs)	Mean (μs)	Pub Bandwidth (Mbit/s)
100	343.06	34.46	64.24	54.05
1000	266.82	34.81	52.09	61.29
10000	357.40	33.43	43.15	55.50
1000000	399.11	26.02	40.98	61.54

puting scenario.

4.4 Case Study, Results and Performance

Fig. 4.8 shows two scenarios for the test results. *Droop_ 0* IED in MG-1 is the main research target. **Scenario 1** is used to evaluate the islanded microgrid clusters and produce the steady state for **Scenario 2**. **Scenario 2** conducts a man-in-the-middle cyber attack to manipulate secondary frequency regulation command and cause catastrophe across the cluster. Scenarios 2 is similar to the real-world industroyer cyber attack in 2016 and industroyer2 attack in 2022 on Ukraine power grids [121], which hijacked supervisory control and data acquisition (SCADA) systems and sent dangerous commands to IEC-104

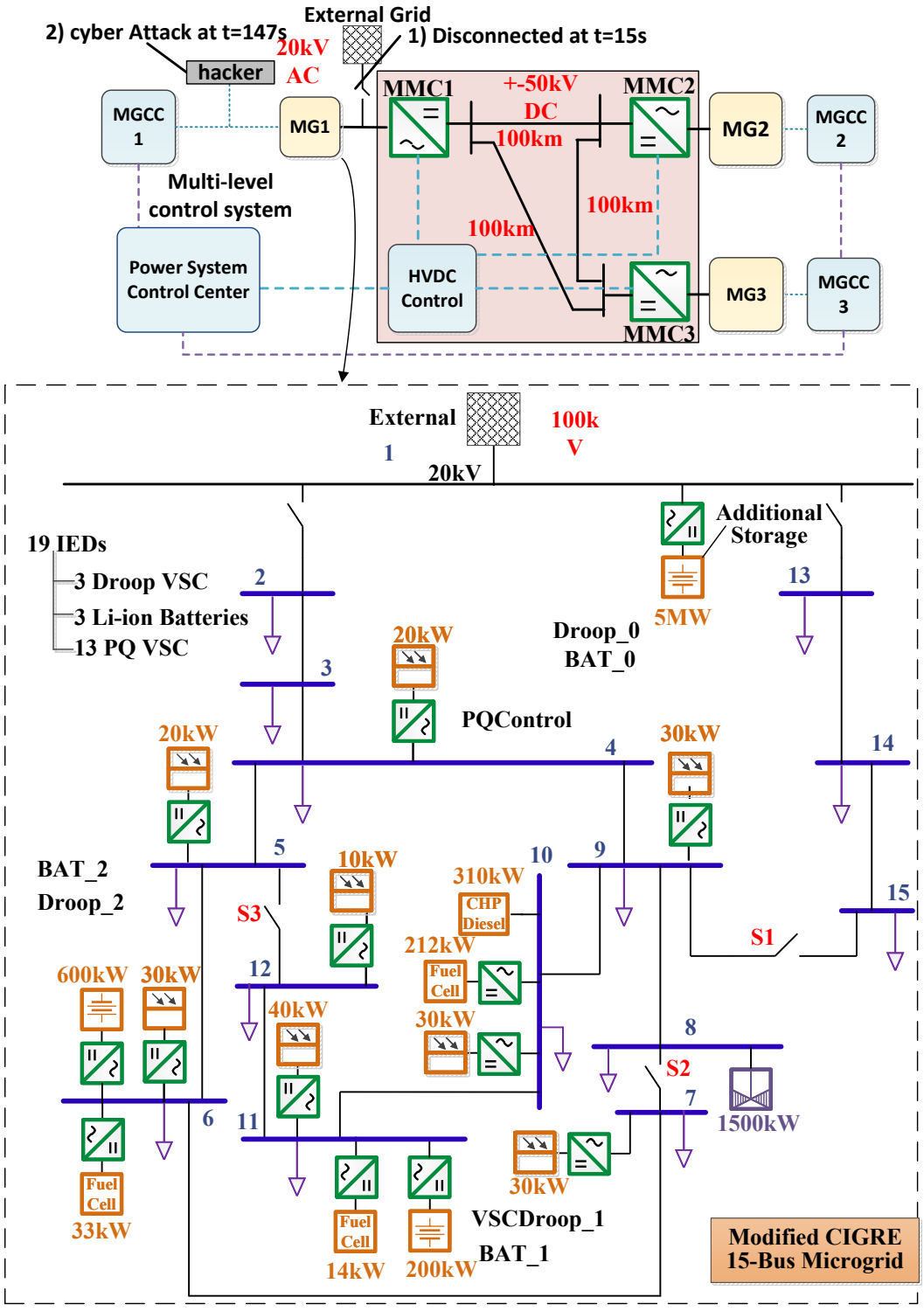

Figure 4.7: System topology and the detailed configuration of the 711-node microgrid cluster.

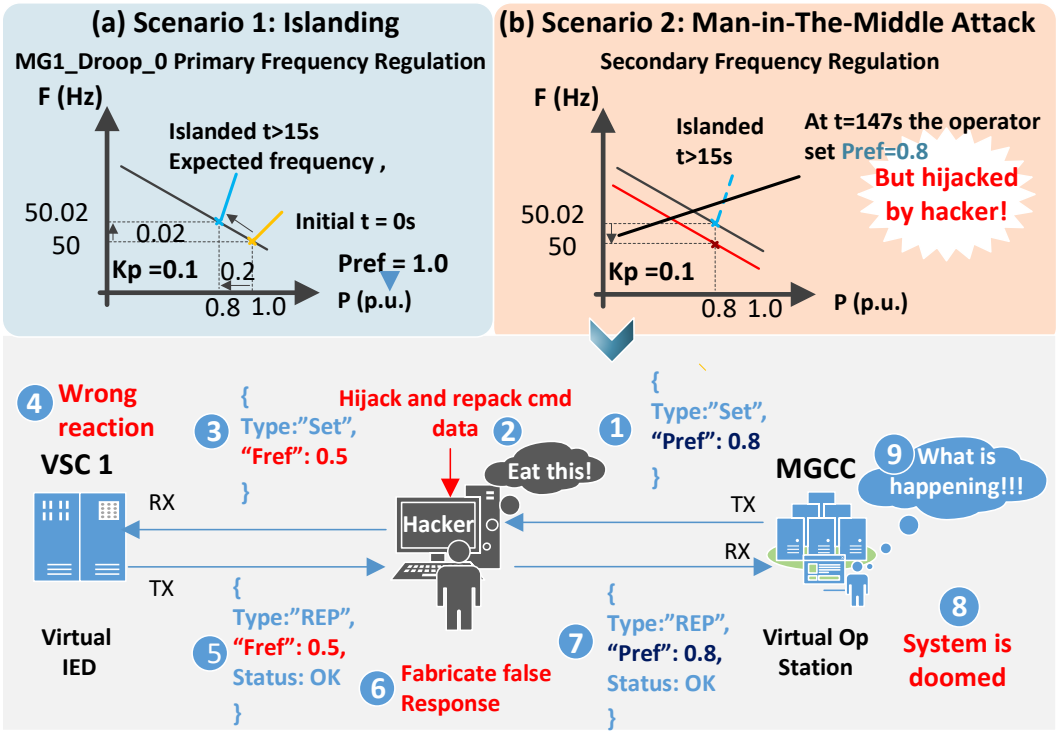

Figure 4.8: Configurations and expected results of test scenarios: (a) **Scenario 1**: islanding operation; (b) **Scenario 2**: Man-in-the-middle cyber attack.

RTUs and IEDs.

Fig. 4.9 shows the setup of the real-time hardware platform introduced in Fig. 4.6. The three NVIDIA® Jetson AGX Xavier embedded computers with real-time Linux installed are used to simulate physical microgrids, the corresponding MMC station, and vIEDs. The Xilinx® VCU118 board is used to handle fast signal IO to support hardware-in-the-loop functions. The PC server runs cyber services such as virtual MGCC, IEC 60870-5-104 clients, and cyber simulation tools.

Fig. 4.7 shows the microgrid cluster connected by a multi-terminal DC system, which forms a 711-node power system with 60 vIEDs to evaluate the proposed simulation platform's functionalities and performance. The microgrid is a 15-Bus power distribution system derived from the CIGRE report [122] and pandapower [123] case files; loads are reduced to 10% and a 5MW Li-ion battery storage is added to Bus-1 to ensure the ability of islanded operation. The microgrid has 16 VSC stations and they are all modeled by the average-value

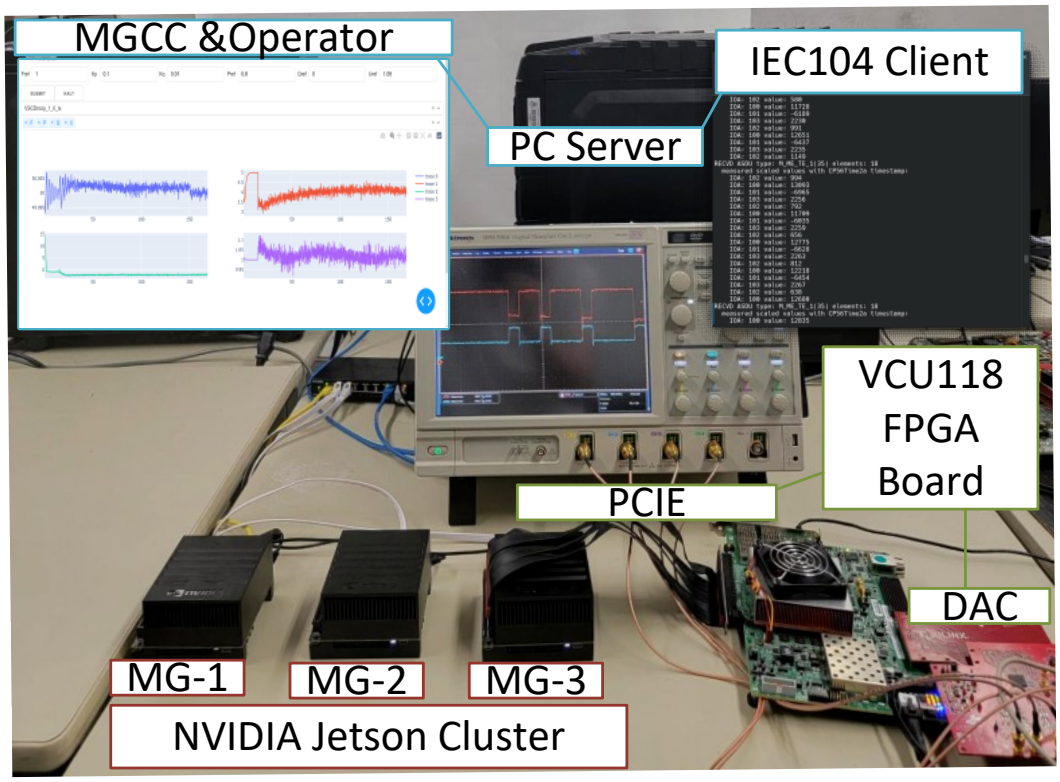

Figure 4.9: Distributed hardware setup of proposed real-time ECS-Grid platform.

model to reduce the complexities of control and simulation. Each VSC station has a VSC controller and a vIED acting as a remote terminal unit (RTU) to the VSC station. Each battery storage has an extra vIED to control the battery charging. The distributed power sources are controlled as PQ nodes which have fixed power generations, while the storage stations are controlled by a droop controller to auto-balance the system and provide a stable frequency. The loads are modeled by fixed RLC components for convenience. Three modified CIGRE 15-Bus microgrids are connected to the three-terminal high-voltage direct current (HVDC) system. The ±50kV HVDC system consists of three 51-level 50MW modular multilevel converters (MMCs) and MMC-1 is designated to control the DC voltages. The other 2 MMCs are set to drain 1MW from the HVDC system. The MMCs are modeled by detailed-switching models which means voltage balancing of submodules is needed. In this work, the nearest-level modulation is used for MMC's lower-level controller. The upper-level controllers for MMCs are similar to VSCs in microgrids which control the DC voltages or the power generations.

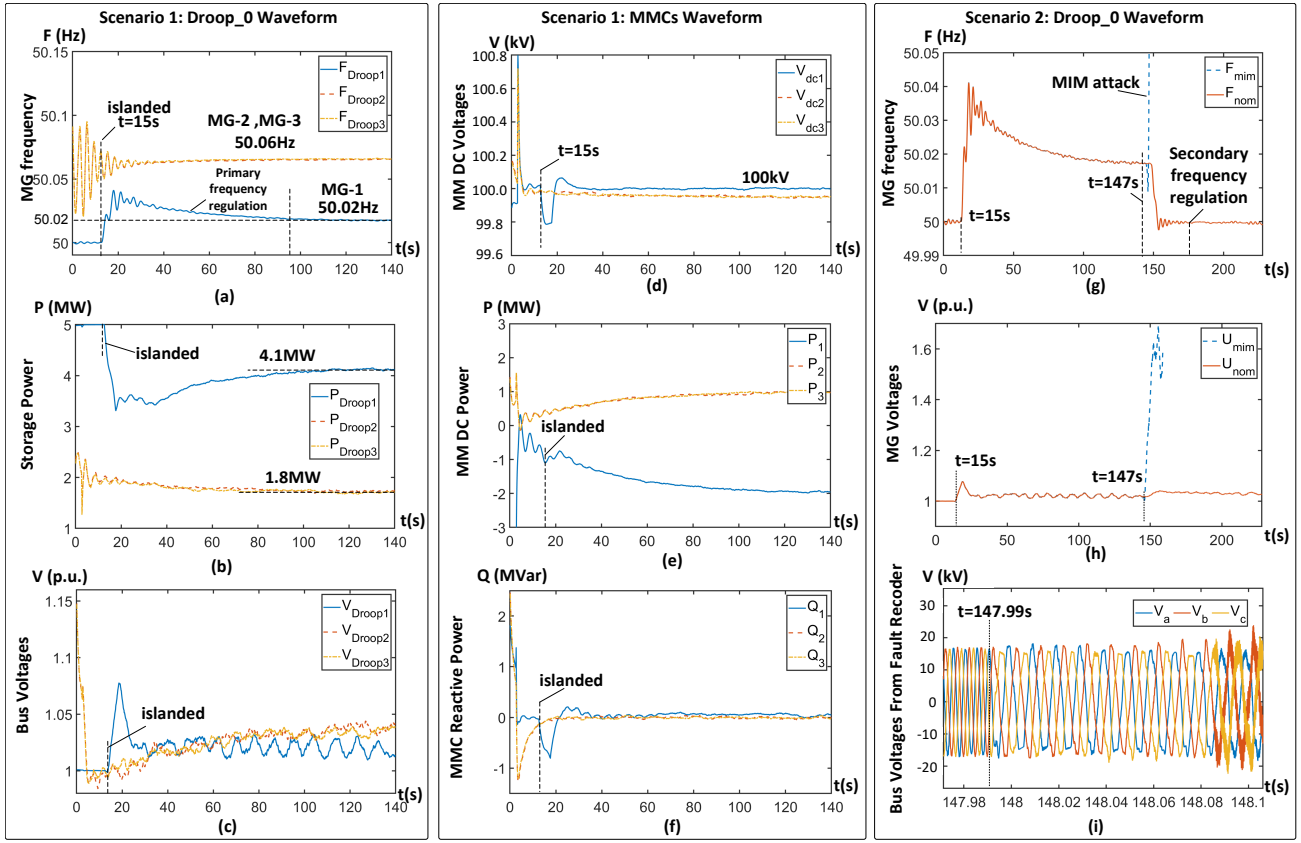

Figure 4.10: Simulation results: **Scenario 1: islanding operation**:(a) Frequency of three *Droop_ 0* stations in each microgrid . (b) Real power of each *Droop_ 0* station. (c) Bus Voltages of each *Droop_ 0* station. (d) MMC DC voltages during the islanding operation. (e) Real power of each MMC station. (f) Reactive Power of each MMC station. **Scenario 2: simulated cyber attack**: (g) Frequency comparison between normal operation and cyber attack situation of *Droop_ 0* in MG-1. (h) *Droop_ 0* Bus voltages comparison. (i) *Droop_ 0* EMT bus voltage waveforms captured by virtual fault recorder.

4.4.1 Results and Performance

The results of **Scenario 1** are shown in Fig. 4.10 (a)-(f). Fig. 4.10 (a)-(c) are the frequency, real power, and bus voltage waveforms of *VSCDroop_ 0* storage station in each microgrid, respectively; when $t < 15s$, the frequency in MG-1 F_{Droop1} is 50Hz since the ideal three-phase AC source is attached to MG-1 Bus-0, while the other microgrids have different frequencies; at $t = 15s$ the ideal source is removed so that a large deviation occurred to F_{Droop1}, while MG-2 and MG-3 have no obvious changes because the DC system can allow asynchronous frequencies; the steady-state value of F_{Droop1} is 50.02 and P_{Droop1} is 4.0MW (0.8 p.u.), which meets the droop control equation $F - F_{ref} = K_p(P_{ref} - P) = 0.02$. This is considered a primary frequency regulation process. Fig. 4.10 (d)-(f) are the DC voltage, real power, and reactive waveforms of MMC stations, respectively; the rated DC voltage is 100kV so that all MMC stations maintained a nominal voltage according to Fig. 4.10 (d), while MMC-1 was impacted by the disconnecting event at $t = 15s$; Fig. 4.10 (e) shows the real power balancing between three MMCs, where MMC-1 provides 2MW and other MMCs drain the planned 1MW from the DC system; the reactive power was set to 0MVar, however, it seems to have large deviation at MMC-1 which may cause the larger voltage fluctuation in MG-1. The results are verified against theoretical analysis and commercial PSCAD/EMTDC®; all data are measured from vIEDs with the interval of $100ms$ and recorded by remote self-made Supervisory Control and Data Acquisition (SCADA) system.

The results of **Scenario 2** are shown in Fig. 4.10 (g)-(i). the dashed line indicates the waveforms under cyber attack while the solid line indicates the normal reactions; since the frequency of MG-1 was 50.02Hz after islanded, the operator sent a command at $t = 147s$ to reset P_{ref} to 0.8 since the current real power is 0.8 p.u., which is a secondary frequency regulation process to restore rated operation point. Fig. 4.10 (g) shows the frequency of *VSCDroop_ 0* in MG-1; under cyber attack situation, the frequency regulation command was intercepted and replaced to $F_{ref} = 0.5p.u.$, which generates very drastic deviations in all measurements from IEDs after $t = 148s$ such as the voltages

Table 4.6: FTRT Performance with Various Communication Intervals

Interval	T_{com}	t_{cps} (s)	r_{ftrt}	Efficiency
2	$100\mu s$	17.88	2.24	0.44
20	1ms	11.02	3.63	0.71
200	10ms	10.24	3.91	0.77
1000	50ms	8.08	4.95	0.97
2000	100ms	8.07	4.96	0.98

System Scale: 711 nodes, 60 vIEDs in total; Simulation duration: $40s$, $\Delta t = 50\mu s$, $t_{phy} = 7.87s$.

in Fig. 4.10 (h). However, these drastic deviations are measured from PLLs and may not reflect the real situation in the physical systems under faulty conditions and that is why real-world power systems also have digital fault recorders to record the EMT waveforms when faults occurred. Fig. 4.10 (i) shows the EMT voltage waveforms of Bus-1 captured by virtual digital fault recorders, which are triggered by the fault detection mechanism in the proposed simulation platform. The EMT waveforms revealed the physical details that happened after $t = 147s$; the system started to react about $1s$ later than receiving the hacked message, and the drastic deviations in measurements may be caused by the high-frequency oscillation which can affect stability. The simulation of Scenario 2 shows the catastrophic consequence of cyber attacks in a vulnerable power cyber network.

Besides the real-time Jetson platform, Table 4.6 shows the performance of the proposed cyber-physical simulation platform on an x86 machine (Intel® Core™ i7 10700k 8c16t@4.7GHz, 32GB DDR4 3000MHz, Ubuntu 20.04, GCC 11.1). The pure physical parallel simulation consumes 7.87s which is 5.08 times faster than real-time, and all cyber-physical simulations also achieved faster-than-real-time (FTRT) performance even with $100\mu s$ ticking interval. For the millisecond-level communication intervals, the cyber-physical co-simulation can achieve high efficiency and the overhead is almost deflectable. The overhead can be further reduced with more concurrency for socket polling, data encoding, and decoding since the communication systems currently execute in

series inside the main simulation loop. The FTRT functionality can enable predictive and preventative control actions in energy control centers.

4.5 Summary

ECS-Grid is a novel data-oriented cyber-physical simulation platform for microgrids under the ECS framework proposed to model the IEDs in a power system with flexible data components and extensible plugin architecture. Furthermore, a modern JSON-like MessagePack-based protocol is proposed for the vIEDs and is capable of completing various tasks needed for cyber-physical transient simulation. The results from the scenarios in the microgrid cluster study case show the accurate system behaviors and real-time or FTRT performance of ECS-Grid. The IED systems and components can be extended to cyber-physical power dynamic or steady-state simulations thanks to the data-oriented design. The data-oriented ECS-Grid can inspire the renovation of industrial software tools and boost further research of the future CPPS.

Chapter 5

Real-Time Cyber-Physical Digital Twin for Low Earth Orbit Satellite Constellation Network Enhanced Wide-Area Power Grid

5.1 Introduction

[1] This chapter is organized as follows: Section 5.2 presents fundamental knowledge to locating a satellite as well as computing the latencies between communication entities in a satellite network. Section 5.3 introduces the design and implementation of RustSat, SatSDN, and ECS-Grid cyber-physical digital twin. Section 5.4 shows the results of wide-area AC-DC grid scenarios with different satellite network setups.

Low Earth Orbit (LEO) satellite constellation networks, which consist of many small satellites that work together to provide a range of communication services, have a great growth in recent years [124], [125]. The most famous LEO satellite constellation network is Starlink from SpaceX^{TM}, which has more than 3000 satellites in space to provide internet connections to nearly all parts of the globe [126], including remote areas where traditional infrastructure is difficult to install or maintain. Unlike traditional satellite communications, LEO satellite constellation networks can provide high bandwidth, low-latency

[1]This work is under review: **T. Cheng**, T. Duan and V. Dinavahi, "Real-time cyber-physical digital twin for low earth orbit satellite constellation network enhanced wide-area power grid," *IEEE open j. Ind. Electron. Soc.*, pp1-10, 2024.

and low-cost network access thanks to their low orbit altitudes. The antenna is very small and easy to install. Due to these advantages, LEO satellite networks bring many new opportunities to the power and energy industry since modern power grids are heavily reliant on communication infrastructure for operation and control.

However, in the discourse on the ideal communication backbone for power system operations, a predominant view heralds optical fiber networks as the ultimate and infallible choice, dismissing any role for wireless communication as unacceptable. Yet, the reality diverges from this conviction. On November 30, 2023, the North American Electric Reliability Corporation (NERC) published a report detailing an incident of real-world communication loss. This event occurred during the transition from a microwave communication system, which was undergoing an upgrade, to an optical fiber communication path [127]. This kind of fault is very typical in daily power grid operations. In this report, the primary differential protection units of a 230kV transmission line are set to communicate via microwave communication channels, indicating that wireless communication is still in service even for real-time differential protection. The optical fiber pathway was interrupted due to ice slowly building up inside the underground cables, which takes 12 hours before complete failure. In this case, the LEO satellite networks, which have thousands of satellites in space for connections, can serve as a swift and resilient backup to traditional communication methods, thereby minimizing the chance of losing critical monitoring data [128]. Similarly, wide area measurement system (WAMS) [129], which uses distributed phasor measurement units (PMUs) to conduct real-time monitoring, control, and analysis of the power grid's performance across a wide geographic area, can benefit from the fast and reliable LEO satellite constellation network to communicate especially when the network service is not reliable such as in rural areas and offshore fields [130]–[132]; This can also bring significant improvements to the power system automation in remote and underdeveloped regions suffering from the "Digital Divide", which barely have any modern communication infrastructures [133]. However, although the applications of LEO networks on power system mea-

surement and control have been discussed, none of the existing research could be able to construct a cyber-physical power system (CPPS) simulator for LEO satellite networks.

Currently, the CPPS simulation can be realized with the network interfaces provided by commercial real-time EMT power system simulators [110], [112]. Alternatively, there are also many self-developed CPPS simulation platforms [36], [108], [134], [135], each with its characteristics and advantages. However, none of these platforms have considered integrating satellite network simulation. LEO constellation networks, which comprise numerous fast-moving satellites [136], are distinct from static ones such as geosynchronous satellites, bringing the following challenges: (1) The swift orbits of LEO satellites demand precise state prediction and real-time simulation for high-performance implementation; (2) A interdisciplinary digital twin built on LEO constellations requires a flexible, scalable solution that connects physical and cyber layer seamlessly to support varied research needs; (3) Co-simulating satellites, cyber networks, and power systems necessitates balancing practicality and accuracy. In summary, the simulation of CPPS with LEO satellite constellation networks requires the integration of expertise from astronomy, power engineering, communication systems, and software engineering, demanding a new digital twin solution for interdisciplinary collaboration.

Aiming to address the above-mentioned challenges, this work proposes a modular data-oriented digital twin solution to perform a cyber-physical co-simulation of a wide-area power grid with the integration of LEO constellation networks. The solution consists of three modules: *RustSat*, *SatSDN*, and *ECS-Grid*. RustSat is the main actor in realizing the LEO satellite constellation network digital twin, which addresses the first challenge; SatSDN, which is based on MiniNet, serves as the cyber-layer between RustSat and ECS-Grid to provide a cyber-layer digital twin for the dynamic satellite constellation network. ECS-Grid is a data-oriented EMT simulation platform that can provide digital twins for Intelligent electronic devices (IEDs) [134]. The three modules are wired together via high-performance ZeroMQ and allow flexible distributed deployments. More specifically, the major innovations of this solution are:

1. High-Performance Data-Oriented Design: The RustSat and ECS-Grid leverage Bevy [48], a Rust-based data-oriented Entity-Component-System (ECS) framework. This framework optimizes data storage in cache-efficient archetype arrays, enabling batched operations on uniform data structures. This data-oriented approach enhances both flexibility and scalability, surpassing traditional object-oriented programming design. Digital twins seamlessly merge physical and virtual data throughout a product's lifecycle, emphasizing the significance of both data streams in their design [137]. The ECS-based architecture aligns with the data-centric philosophy of digital twins, fostering various data flows. Thus, the ECS-based design effectively complements the data-centric nature of digital twins.

2. Flexible Plugin Architecture: Based on the ECS framework, all RustSat and ECS-Grid functionalities and systems are implemented by plugins. These plugins can be added to the main software application to extend its capabilities without altering the core structure and functionality of the application, enabling users to customize and extend the software's functionality to fit their specific needs. Compared to traditional simulation tools [138], [139], the proposed architecture brings superior modularity, scalability, and flexibility to face the interdisciplinary complexity challenge, enabling efficient deployment and expansive growth within the realms of digital twin implementations.

3. Practicality and Authenticity in Co-Simulations: Being a real-time digital twin solution, there's no place for extra synchronization mechanisms beyond real-world scenarios for the physical and cyber layers. RustSat and ECS-Grid are intrinsically integrated with cyber communication plugins, enabling cyber-physical simulation. The cyber network simulation is managed by SatSDN, built on MiniNet. For simulating LEO satellite networks, MiniNet's software-defined network (SDN) technology separates the network control plane from the data plane. This dynamic network topology adjustment is essential for effective communication in

satellite constellations [140], [141]. ECS-Grid's virtual intelligent electronic devices (IEDs) seamlessly communicate with MiniNet's virtual network gateway nodes via customized network configuration.

The proposed data-oriented approach is the first to address the physical data acquisition, model simulation, and multi-system collaboration, representing a comprehensive framework for a real-time LEO satellite network digital twin for cyber-physical power system analysis. The proposed framework is evaluated on the modified IEEE 118-Bus system + MMC wide-area AC-DC test system together with Starlink LEO, GPS, and GEO satellite networks. The evaluation results demonstrated superior performance and accurate results verified against well-known software.

5.2 LEO Satellite Simulation Fundamentals

Different from traditional network simulation, the LEO satellites are moving very fast in space, leading to a highly dynamic network topology. Therefore, it is vital to simulate the trajectories of LEO satellites to build a realistic cyber layer from them.

5.2.1 Locating LEO Satellites

The LEO satellites orbit the Earth when they are attracted by the Earth's gravity and move in a circular or elliptical path around it. Therefore, it is vital to know the accurate position and trajectory to establish a connection between devices in an LEO satellite network. This information is essential to analyzing a satellite constellation network. Some basic astronomical knowledge is required to obtain this fundamental information.

Orbital elements are the parameters required to uniquely identify a specific orbit [142]. As shown in Fig. 5.1, there are the six Keplerian elements that define the size, shape, and orientation of the orbit. However, the satellite orbits are not fixed due to the gravity and rotation of the Earth, and the drags of the atmosphere. The SGP4 (Simplified General Perturbations 4) algorithm in [143] is a method of simulating the orbit of a near-Earth satellite and using the

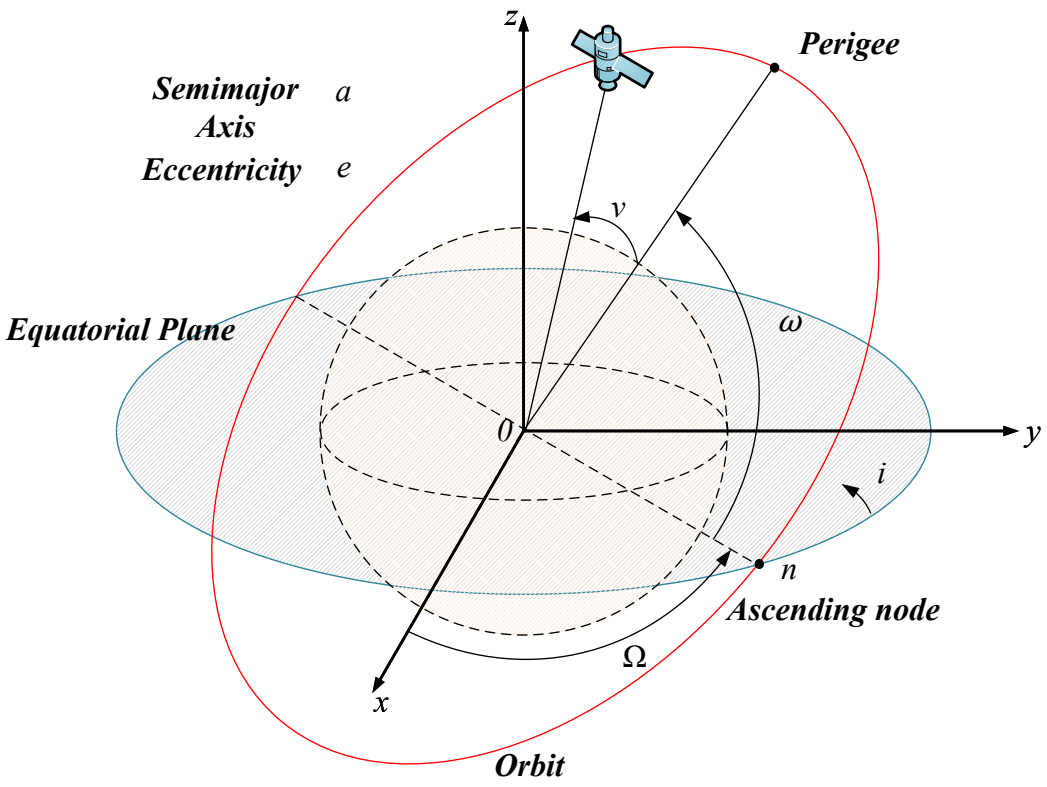

Figure 5.1: Classic orbital elements: Semi-major Axis a, Eccentricity e, Argument of perigee ω, True anomaly $\boldsymbol{v}$, longitude of ascending node Ω, and inclination i.

known orbital elements to predict the position and velocity of the satellite at a future time. It is a relatively simple, fast, and accurate method, and has been widely used for this purpose since the 1970s. The algorithm was designed to work with Two-Line Element (TLE) data format orbit elements from the North American Aerospace Defense Command (NORAD). It involves a significant number of constants, equations, and symbols to get an accurate result. In 2006, the SGP4 algorithm was revised by [144] and modernized source codes of SGP4 were released. The algorithm details and related publications can be found on the Celestrak website [145]. Now, there are many open-source implementations in almost every major programming language. In this work's implementation, the SGP4 library from [146] is used to predict the accurate real-time position and velocity for further analysis.

The satellite position coordinates of SGP4 are typically given in a geocen-

tric coordinate system called True Equator Mean Equinox (TEME) system which is an Earth-centered inertial (ECI) system that is centered at the center of mass of the Earth and is a non-rotating system referred to the celestial sphere. However, the ground communication stations on the Earth's surface use a geodetic coordinate system and rotate with Earth so that conversion from TEME coordinates to geodetic coordinates is required to compute the relative distance and latency between ground stations and satellites.

The conversion takes 3 steps:

1. Convert the satellite coordinate epochs in UTC format into Greenwich Mean Sidereal Time (GMST). This is used to calculate the satellite's position relative to the Earth's surface. The GMST is an hour angle that measures the Earth's rotation with respect to the mean position of the sun. The sidereal time can be computed by [142]:

$$\begin{aligned} H &= 67310.54841 \\ &\quad + (876600.0 * 3600 + 8640184.812866)T_u \\ &\quad + 0.093104T_u^2 - 6.2 \times 10^{-6}T_u^3 \end{aligned} \tag{5.1}$$

$$,T_u = \frac{JD_{ut1} - 2451545.0}{36525}, \tag{5.2}$$

$$\theta_{gmst} = \frac{2\pi H}{86400}, \tag{5.3}$$

where H is the GMST in seconds; θ_{gmst} is the GMST in radians; JD_{ut1} is the Julian Day (JD) converted from Universal Time 1 (UT1) epoch t_{UT1}. UT1 epoch can be computed from the common Coordinated Universal Time by $t_{ut1} = t_{utc} + t_{dut}$, where $-0.9s < t_{dut} < 0.9s$ is caused by the difference between Earth rotation and atomic clocks and controlled by adding leap seconds to UTC stamps.

2. Convert the TEME (or ECI) coordinates to Cartesian coordinates under a rotating Earth-centered, Earth-Fixed (ECEF) reference frame. This conversion relies on the previous θ_{gmst}. If the movement of the equatorial plane and poles of the Earth are ignored, the conversion can be done by multiplying a rotation transform matrix.

3. Convert ECEF Cartesian coordinates X, Y, Z to longitude λ, latitude

ϕ, and altitude h under WGS84 (World Geodetic System 1984) used by GPS so that the satellites can be easily marked on a world map along with ground stations. The conversion of longitude λ is straight-forward by

$$\lambda = \text{atan2}(Y, X). \tag{5.4}$$

The conversion of latitude ϕ, and altitude h starts with the WGS84 GPS ellipsoid frame because the Earth is not a perfect sphere. However, the computation requires Netwon-Raphson iterations. There are various iterative formulas to solve for latitude. Here, a closed-form formulation proposed by Bowring is presented [147]. First, calculate intermediate values:

$$\begin{aligned} &p = \sqrt{X^2 + Y^2}, \\ &b = a(1-f), e' = \frac{a^2 - b^2}{a^2}, \tan\beta = \frac{Z}{p(1-f)}, \end{aligned} \tag{5.5}$$

where $f = 1/298.257223563$ is the flattening rate and $a = 6378137.0m$ is the semi-major axis defined by WGS84 ellipsoid frame. $\tan\beta$ is an initial guess that can achieve reasonably high precision without iteration, and β is defined by

$$\beta = \text{atan}\,\frac{(1-f)\sin(\phi)}{\cos(\phi)}. \tag{5.6}$$

With the initial guess $\tan\beta$ the latitude ϕ and altitude h can be computed by:

$$\phi = \text{atan}\,\frac{Z}{p(1-e')}, \tag{5.7}$$

$$h = p\cos(\phi) + Z\sin(\phi) - a\sqrt{1 - e'(\sin^2(\phi))}. \tag{5.8}$$

The closed-form formulation can give very high accuracy at the initial guess and the maximum error (0.0018", 5 millimeters) is achieved at $h = 2a$ [147]. (5.8) is a height equation with higher accuracy from [148].

After obtaining the WGS84 coordinates, the satellite can be visualized on a flat 2D map using Web Mercator projection [149] used by online maps. The aforementioned equations and methods covered all fundamentals of developing satellite tracking software.

5.2.2 Distance between Satellite and Ground Stations

For each satellite, its circular orbit is characterized by the altitude H, respectively with the radius r interrelated as $r = R + H$, where R is the radius of Earth. Given the location of a ground station (GS), the distance between the GS and the satellite is of importance to compute the communication delay. It could be seen in Fig. 5.2 that, the satellite (denoted as point S), GS (denoted as G) and geocentric (denoted as O) make up a triangle, thus the distance between S and G (denoted as d_{sg}) can be obtained if the included angle (θ) between the line segments SO and GO is known:

$$d_{sg} = \sqrt{r^2 + R^2 - 2rR\cos\theta}. \tag{5.9}$$

In fact, the included angle θ can also be computed based on the arc between P_s and G, where point P_s is the projection of the satellite S at the Earth ground:

$$\theta = \frac{\widehat{P_sG}}{R}. \tag{5.10}$$

The *arc* $\widehat{P_sG}$ can be obtained based on the longitude and latitude of P_s and G. In addition to the distance between the satellite and the ground station, the elevation angle between the satellite and the ground station's horizon plane (denoted as ε_s, ranging from 0° to 90°) is also important since a satellite could only communicate with the ground station between the acquisition of the satellite (AOS) event and the loss of the satellite (LOS) event, as shown in Fig. 5.2. Considering the barriers (natural or artificial) that may impact the visibility, the horizon plane with an appropriately designed elevation angle ($X°$, denoted as ε_0) is applied. Therefore, the necessary condition for a satellite to communicate with a ground station is that ε_s is larger than ε_0:

$$\varepsilon_s > \varepsilon_0. \tag{5.11}$$

Given the location of two ground stations (G and G_2 shown in Fig. 5.2), the end-to-end packet transmission distance through the low Earth orbit satellite constellation network is of concern to estimate the end-to-end transmission delay. In this work, two assumptions are proposed: first, a ground station

always sends its data packet to the satellite with minimum d_{sg} among all the satellites that meet the requirement (5.11); second, two satellites with low distances directly send the packet to each other without the usage of relay nodes. Based on (5.9), at each time slot, the location of the satellites with minimum distances to G and G_2 can be found, denoted as S and S_2, as shown in Fig. 5.2. Based on the above assumptions, the data transmission distance between S and S_2 is their geometric distance d_{ss2}. Therefore, the end-to-end data transmission distance between G and G_2 is estimated as $D(G, G_2) = d_{sg} + d_{ss2} + d_{s2g2}$.

5.3 Data-Oriented Digital Twin Architecture of Satellite Network-Enhanced Power Grid

Based on the mathematical theory provided in Section 5.2, in this section, the data-oriented digital twin architecture is proposed. As shown in Fig. 5.3, the proposed method includes three main subsystems: (1) RustSat: a data-oriented satellite real-time data acquisition and monitoring software implemented with Bevy Engine [48], which computes and provides the satellite positions and trajectories; (2) SatSDN: an SDN simulator to simulate LEO satellites and ground stations communication as SDN devices with Mininet [150] and Docker container [151]; (3) ECS-Grid: a data-oriented, real-time simulator for power system EMT simulation [134], which integrates virtual IEDs to create network traffic into the SatSDN using ZeroMQ.

5.3.1 Motivation for Using Data-Oriented Paradigm

To construct the real-time digital twin of LEO satellite networks enhanced power systems, efficiently integrating astronomy, geography, real-time visualization, computer networking, and power system EMT simulation is a major challenge. It involves interdisciplinary collaboration and the convergence of diverse fields of expertise. The mainstream object-oriented paradigm emphasizes abstractions and object relationships. It encapsulates the data with additional relationships and interfaces to achieve the polymorphism, which also adds un-

necessary complexities and lowers the performance for data-centric simulation tasks. Moreover, traditional object-oriented paradigms heavily rely on inheritance relationships to construct program frameworks. However, pre-defined classes and interfaces cannot anticipate the comprehensive, diverse, and continuously evolving simulation requirements of the digital twin era. Therefore, a more flexible, scalable, and high-performance software design architecture is needed.

In contrast, the data-oriented design paradigm focuses on optimal data layout and data flow processing and tends to avoid unnecessary abstractions of data [152]. The ECS frameworks are the typical representatives of the data-oriented paradigm. In ECS frameworks, entities are defined by dynamic data component combinations instead of static class inheritance; data and program logics are isolated, so users can focus on the data processing without the fear of breaking application interfaces; the data are stored in the structure-of-array layout instead of the typical array-of-structure layout applied in object-oriented designs, which can avoid cache misses and lead to higher performance. Due to these advantages, Gazebo, the renowned robotic simulation platform from Open Robotics, has adopted the ECS paradigm as its backbone [45].

In summary, the data-oriented ECS paradigm and the requirements for data flow and data fusion emphasized by digital twins are in perfect alignment. The data-oriented ECS paradigm is considered the best choice to build interdisciplinary digital twins, the high flexibility, scalability, and real-time performance.

5.3.2 RustSat: Data-Oriented Satellite Digital Twin

RustSat handles all astronomical equations introduced in Section 5.2 and provides satellite information for cyber network simulation. It is written in Rust language and uses Bevy Engine as the fundamental framework. Bevy Engine is a data-oriented ECS game engine built upon Bevy ECS. The Bevy ECS framework mainly uses archetype tables as its storage, which is designed to be fast and easy to use, with a focus on performance and modularity.

The ECS framework, such as Bevy ECS, is composed of three main ele-

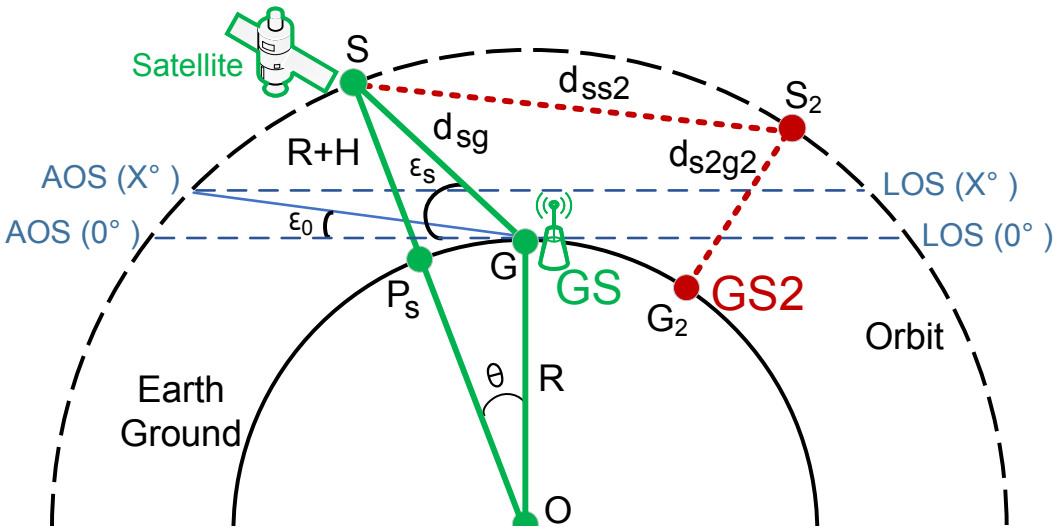

Figure 5.2: Illustration of the satellite orbit and location of the ground station.

ments: *Entity*, *Component*, and *System*. An *Entity* is formed by combining components. *Components* store data exclusively, similar to C programming's *struct*. *Systems*, akin to procedural programming's procedures, are functions to query and process specific *Components*. In ECS, *Systems* handles program logic and algorithms, while the *Component* arrangement triggers relevant *Systems* for *Entity* processing. ECS's hallmark is its reliance on data combinations to define program functionality, which greatly benefits data-centric digital-twin scenarios.

Fig. 5.4 (a) shows the major entity archetypes in RustSat; each archetype can be seen as a data table representing a unique combination of various *Components*; each *Component* type, which is represented as a row in the data table, is stored in a contiguous array; each column represents an entity of this archetype. By using this structure, satellites, ground stations, and communication data links can be represented by a combination of basic simple data *Components* at run-time, which improves the reusability and flexibility.

Fig. 5.4 (b) shows the example of three *Systems* in RustSat. The *SGP4 Algorithm System* reads orbital elements from *SGP4Constants* and writes results to *TEMEPos*. Therefore, this system will only process *Satellite Entities* because the other archetypes do not have these *Components*. In contrast, the

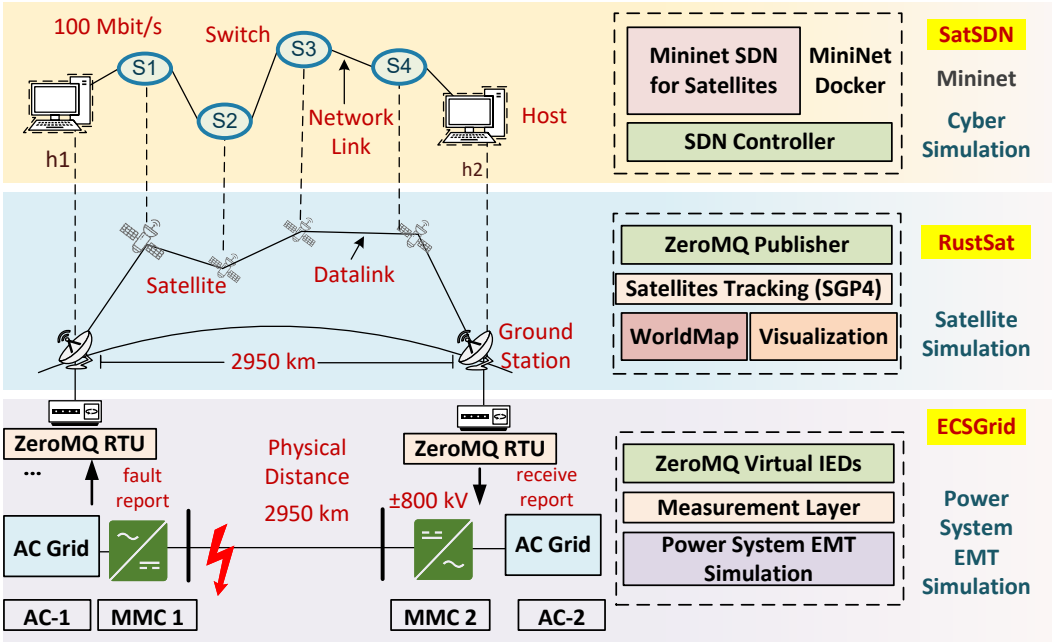

Figure 5.3: Three modules of the satellite-network-based cyber-physical power system digital twin: RustSat, SatSDN, and ECS-Grid.

RenderSystem processes all three archetypes because they all have the components for visualizations. The *Systems* process rows of archetype tables so that it is more cache-friendly and easier to be vectorized.

Fig. 5.4 (c) shows the data flow of RustSat. The *Systems* are grouped into different plugins such as *SGP4Plugin* and *SatRenderPlugin*; they are inserted into a directed acyclic graph (DAG) and the independent *Systems* such as *ZMQPlugin* and *SatRenderPlugin* can be scheduled to run concurrently.

In summary, both object-oriented programming and data-oriented programming can be viewed as assembling building blocks to construct programs, but the building blocks used in each approach differ. In object-oriented programming, the building blocks are classes and objects, which are often determined at compile-time. In data-oriented programming, the building blocks are data components and data flows assembled by system functions, which can be rearranged at run-time. The data-oriented paradigm is suitable for digital twins that prioritize efficient and flexible data flows as the primary factor in program design and implementation.

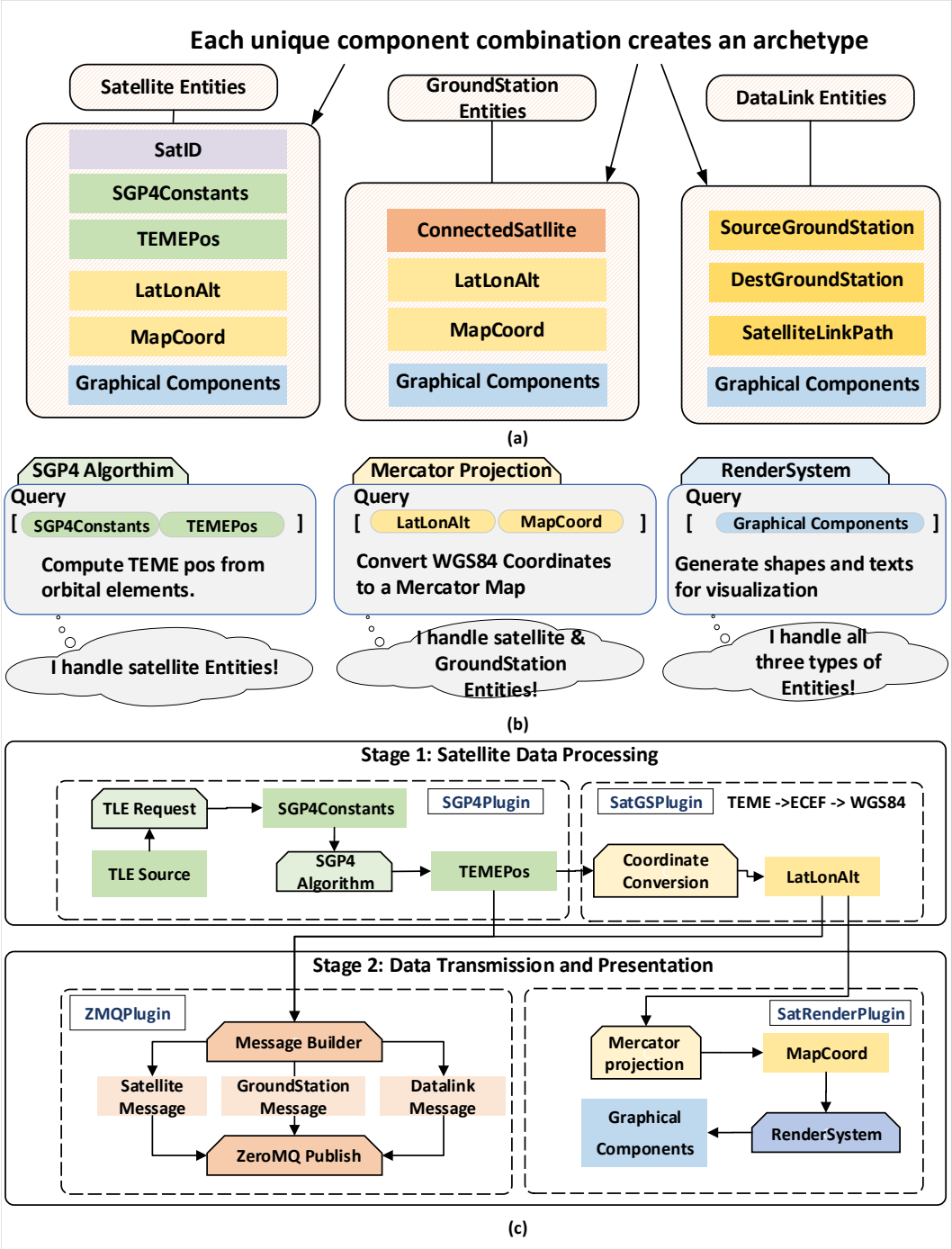

Figure 5.4: Data-oriented design and dataflow of RustSat: (a) archetypes of simulation entities in Bevy ECS; (b) fundamental systems to deal with data components; (c) overall data flow of the RustSat program.

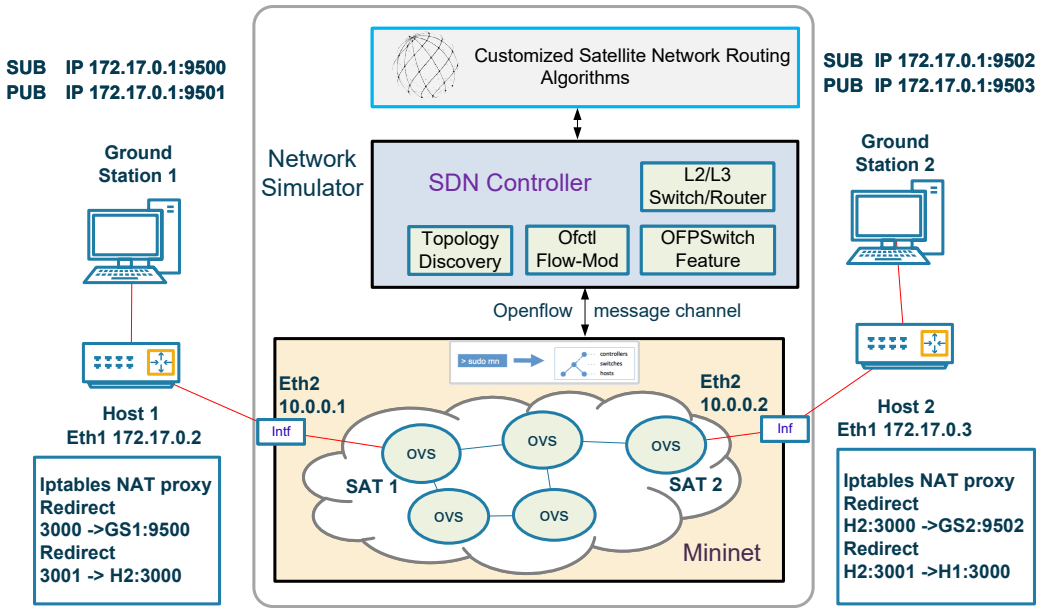

Figure 5.5: Virtualized Mininet SDN network configuration for SatSDN using Docker network addresses.

5.3.3 SatSDN: Containerized Mininet SDN Simulator for Cyber Network Digital Twin

The cyber network simulation is not as simple as a data processing program. It involves multiple software systems and hardware such as the operating system, drivers, network protocols, switches, routers, etc. Therefore, currently, it is not suitable for the ECS framework to do this job. Fortunately, there are several mature open-source network simulation platforms such as OpNet, NS-3, and MiniNet, which can form a solid basis for simulating satellite networks.

SatSDN, built upon Mininet [150], utilizes Mininet to swiftly create virtual networks, deploying kernel, switch, and application code on a single machine (virtual machine, cloud, or native) through simple scripts. Mininet employs a user-friendly Python library to generate virtual ethernet interfaces and topologies, which seamlessly interface with actual Linux kernels, enabling network simulators to incorporate real-world tools. Fig. 5.5 shows the virtual network layout for SatSDN, with Mininet's virtual devices operating within a Docker container. Ground stations communicate with Mininet's virtual hosts via a Docker local network, and Host 1 and Host 2 possess dual interfaces and ad-

dresses, using *iptables* proxy rules to proxy real-world network traffic into the satellite SDN. The ZeroMQ PUB/SUB socket setup is shown in Fig. 5.5 The satellites, resembling SDN switches, are built from RustSat's real-time data. The SatSDN Python app processes satellite position, distance, and communication latency data from RustSat. It dynamically calculates network data link latencies to reflect the satellite movements. Mininet's network data links can additionally simulate bandwidth and packet loss if further satellite details are accessible.

5.3.4 ECS-Grid: Data-Oriented Cyber-Physical Power System EMT Simulation

The theories of EMT simulation and ECS-Grid has been introduced in Chapter 2 and Chapter 4. This subsection will focus on the data-oriented implementation of MMCs from Chapter 2 Section 2.5 under ECS paradigm.

A three-phase MMC is represented as a Norton equivalent circuit as shown in Fig. 5.6 (a). In the new data-oriented ECS-Grid implementation, the three-phase MMC is composed of multiple entities with hierarchical relationships shown in Fig. 5.6 (b). The submodule components are assigned in MMC arm entities as shown in Fig. 5.6 (c).

The admittance matrix of all switching states are enumerated and cached in before simulation since all HBSMs in this *MMCArmBundle* share the same topology and parameters The choke inductor and voltage source equivalent of HBSMs can be expressed by the equivalent circuit shown in Fig. 5.6 (c). It is clear that the equivalent circuit can be further simplified to Norton equivalent circuit which determines the final admittance and current injection in the *CurrentSourceBundle* representing admittance and equivalent current sources used for global circuit solver.

In this way, a single-phase MMC can be established by composing two arms, and a three-phase MMC is a composition of three single-phase MMC or 6 MMC arms. The hierarchical entity relationships make it easier to track and control the whole three-phase MMC with multiple entities. The three-phase MMC entity can spawn all children entities and setup controllers and

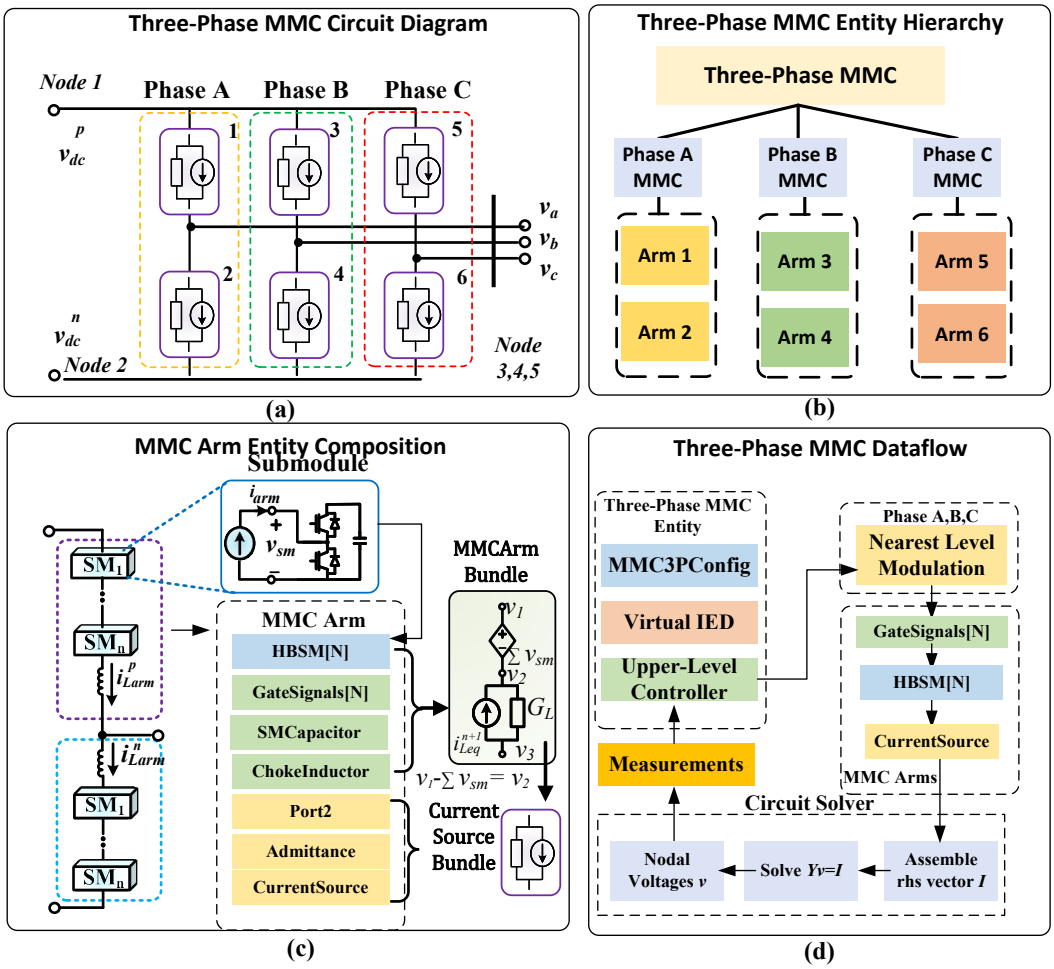

Figure 5.6: ECS implementation of a three-phase MMC in ECS-Grid: (a) equivalent circuit of a three-phase MMC; (b) entity hierarchy of a three-phase MMC; (c) implementation and entity composition of an MMC arm entity; (d) data flow of the three-phase MMC entity.

connections automatically based on information in *MMC3PConfig* before the simulation started. The data flow of these MMC entities is shown in Fig. 5.6 (d). This work employs nearest-level modulation for the lower-level controller of MMC submodules, connecting upper-level power or voltage controllers to three-phase MMC entities.

The virtual IEDs are also installed on the MMC entities. They can use various communication protocols via plugins, while the ZeroMQ protocol is used by default. The ECS-Grid provides network interfaces for IEDs. However, its primary focus is on the power system side, lacking the ability to investigate

the network routing and data link latencies.

The SatSDN provides a practical solution to simulate the LEO satellite network. The vIEDs can be proxied by ground stations' Docker network IP address and ports as shown in aforementioned Fig.5.5, allowing network traffic to be delivered to the simulated satellite network seamlessly. Such a network configuration is similar to the remote communication setup of real-world substations.

5.4 Case Study and Results

5.4.1 Case Setup and Test Environment

In the NERC case study, operators could only watch helplessly as the cable was slowly severed, with no backup options available. However, LEO satellite networks can offer hundreds of backup options in space, accessible with inexpensive and easy-to-install antenna devices for establishing backup channels. If LEO constellation networks can be used for power grid communication, they can maintain connectivity enhance operational efficiency for operators, and avoid such incidents. Therefore, a test scenario is established to show the advantages of using LEO satellite networks for wide-area protection communication and compare their performance with wired communication and other satellite network types.

A synthetic AC-DC test system shown in Fig. 5.7 is established on the proposed platform. The AC System-1 and MMC-1 are located at Calgary ($51.00°\text{N}, 114.03°\text{W}$) and AC System-2 and MMC-2 are located in Toronto ($44.22°\text{N}, 80.11°\text{W}$). The MMC-1 and MMC-2 are connected to AC sources. The parameters of the IEEE 118-Bus system were converted from PandaPower [123]. The physical distance from Calgary to Toronto is around 2950 km. Each MMC has 200 half-bridge submodules (HBSM). The 2950 km transmission line is simulated by the traveling-wave line model and yields a propagation delay of 18.67 ms, which is also the approximated delay of ground communication. The DC system parameters are summarized in Table 5.1. It is intentional to create such a long transmission line since HVDC projects of a similar distance level

exist in China and the LEO satellite network can be better utilized in this long-distance cases. Such HVDC systems are vital for power grid operations and the loss of optical fiber communication of these systems could bring much worse consequences than a 230kV substation in the NERC report.

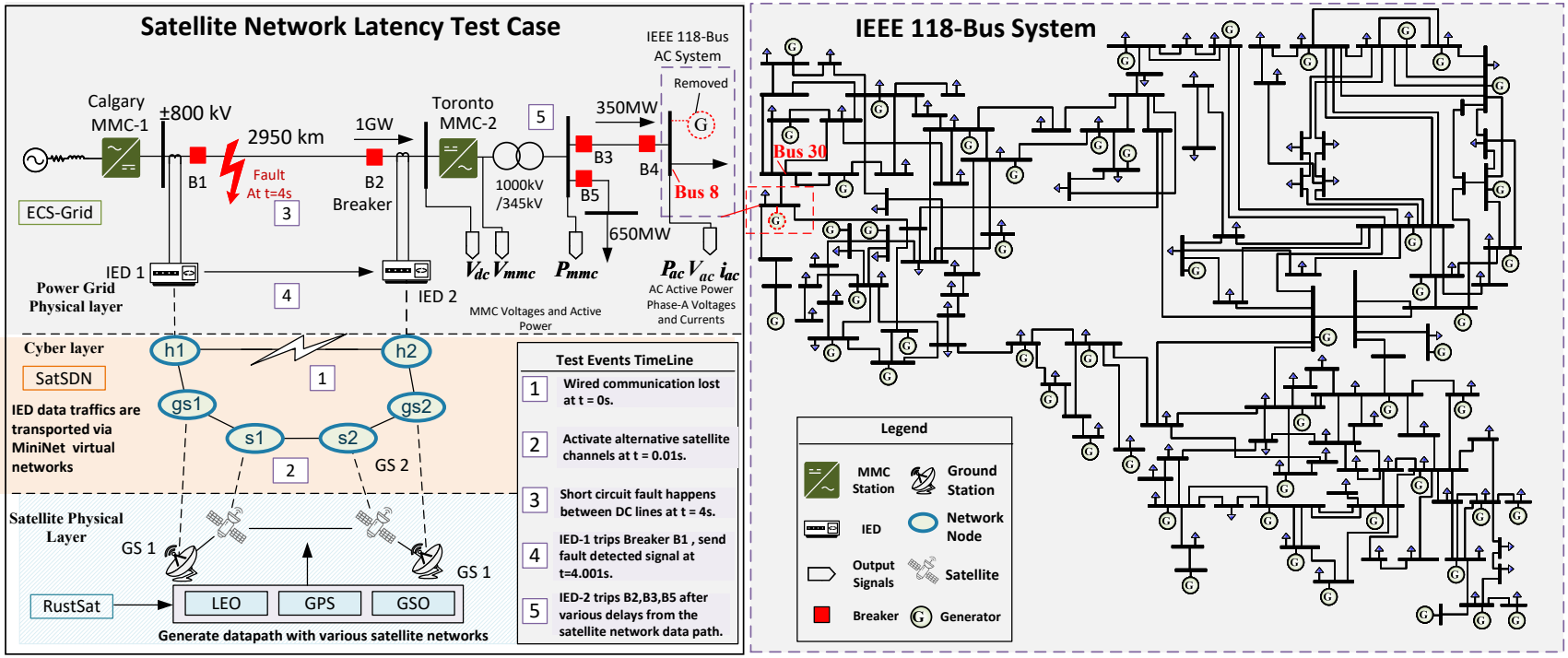

Figure 5.7: The synthetic AC-DC system based on modified IEEE 118-Bus system for evaluating latencies of LEO, GPS and GSO satellite networks.

The Bus-8 in the IEEE 118-Bus system is connected to the MMC-2 as shown in Fig. 5.7. The following modifications are made to connect the MMC-2 to the 345kV Bus-8. First, the generator of the original Bus-8 is removed since MMC-2 serves as the new power source. Second, the MMC-2 is generating 1GW at the steady state which is not suitable for injecting into IEEE 118-Bus system. Therefore, a 650MW load is added at the MMC-2 so that only 350MW active power is injected into Bus-8. The most frightening consequence of communication interruption is the inability to respond to major power system incidents that occur during this period. Therefore, this scenario simulates whether the satellite network system can maintain the original automatic protection system's ability to respond to faults in the event of a wired communication interruption. As shown in Fig. 5.7, there are 5 major events for the test case. In the beginning, the optical fiber connection is lost and the backup satellite network channel is brought up. At 4.0s after the simulation started, a short circuit fault is set to happen at the side of MMC-1, which will be identified by IED-1 installed at MMC-1 within 1ms, and it immediately sends the fault signal to IED-2. Given the loss of wired communication, the transmission of such signals would be redirected through data links facilitated by SatSDN and RustSat. The IED-2 installed at MMC-2 is set to open B2, B3, and B5 breakers to isolate the fault from the MMC-2 and AC grids only according to the remote signal to demonstrate the impacts of different satellite communication delays. The output signals for observation are voltages, active power flow, and currents of MMC-2 and Bus-8.

Fig. 5.8 demonstrates that the satellite TLE data used in RustSat originates from various satellites, including Starlink, GPS, and GEO satellites. It is crucial to acknowledge that the routing method employed by satellites can significantly influence performance. The proposed simulation platform aims to facilitate interdisciplinary collaboration between satellite communication experts and power system engineers, enabling them to design network scheduling algorithms and conduct performance testing evaluations in conjunction with physical simulations of CPPSs. However, the design of SDN routing algorithms falls outside the scope of the simulation platform and is left to future

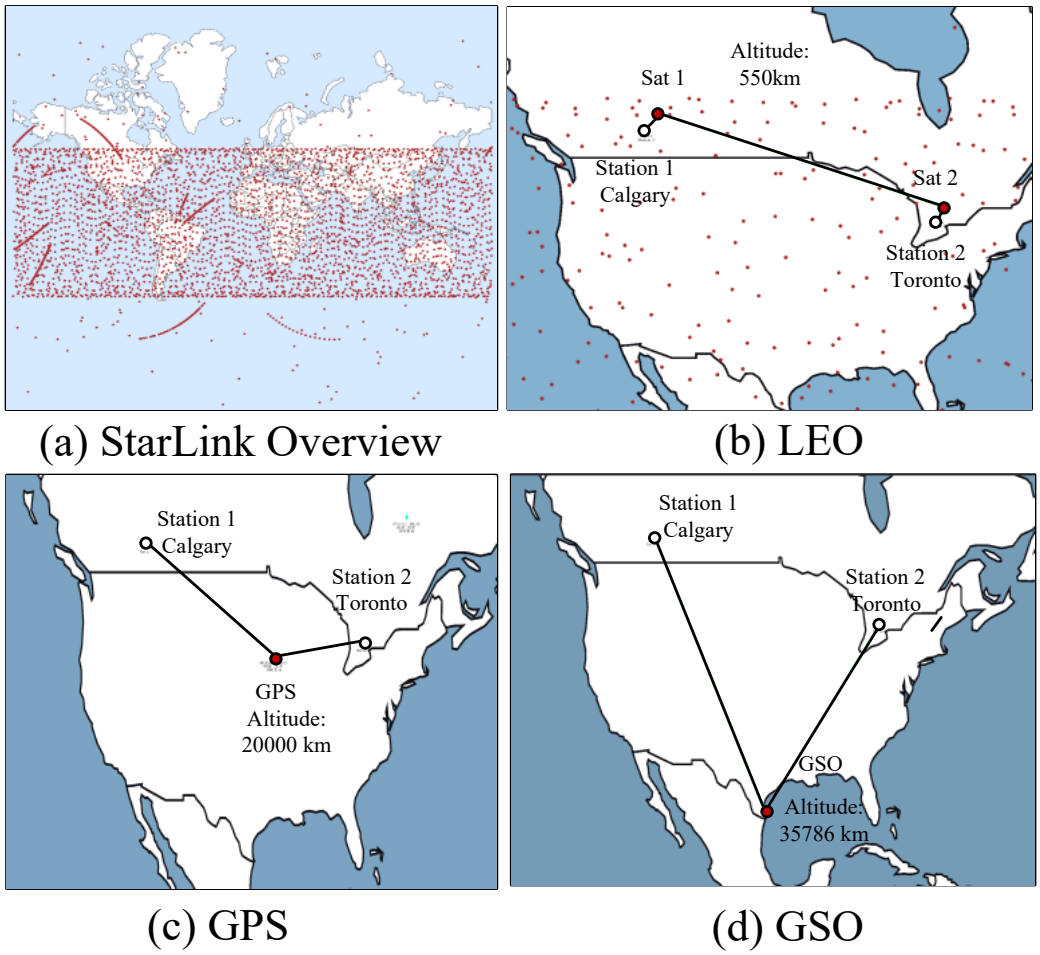

Figure 5.8: Satellite networks visualized in RustSat: (a) overview of Starlink satellites; (b) LEO constellation network connection; (c) network connection with a GPS satellite; (d) network connection with a GEO satellite.

users to implement. In this work, a simple approach to establish data links between ground stations via satellites: each ground station seeks to connect with the nearest satellite, allowing for direct connections. Real-world scenarios may present greater complexity. The three modules: RustSat, SatSDN, and ECS-Grid were all running on the same Intel® Core™ i9 13900K server with the system environment shown in Table 5.2.

5.4.2 Results and Performance

There were 3728 Starlink LEO satellites when the test was conducted. The RustSat takes **1.1ms** to predict the positions of all satellites, while a pure

Table 5.1: MMC and Power System Parameters

MMC Parameters	**Value**
Submodule Type	Half Bridge
DC Link Voltage	±800kV
AC Transformer Voltage	1000/345kV
Rated Power	2GW
Number of Submodules	200
Modulation Method	Nearest-Level
DC Transmission Line Info	
Length	2950km
Resistor	3.44e-6Ω/m
Unit Travel Time	6.32e-9s/m
Propagation Delay	18.6ms
Surge Impedance	369Ω

Table 5.2: Simulation Test Platform Parameters

Component	**Parameter**
Processor	Intel Core i9-13900K 8+16 Cores
Memory	64GB DDR4 3000MHz
Operating System	Ubuntu 22.04 Linux Kernel 5.19
Software Version	Rust 1.68, Mininet 2.3, Python 3.11

Python implementation takes **144ms**. The Rust program is 100x faster than Python implementation and is thus suitable for real-time analysis. The satellite TEME and geodetic results are verified against the open-source Python SkyField library [153].

Fig. 5.9 shows the 5-minute single-trip latency data of Starlink LEO satellites recorded from RustSat. LEO satellites move at high speeds, which requires ground stations to frequently switch their connection to maintain a stable communication link. This can result in jitters in latencies as shown in Fig. 5.9. For Internet communication across the world, the jitters can make quite a large difference in disturbing communication stability [139]. However, the jittering for this study case is not significant because the 2950km distance is not long in the space and the data link is established exclusively for the two ground stations. Further research on the routing of LEO satellite networks

Table 5.3: Satellite Orbit Altitude, Physical Distance, and Average Latency

Satellite Type	Altitude (km)	Distance (km)	Phys. Latency (s)	SDN Latency (s)
GEO	35786	149300.4	0.498	0.501
GPS	20000	40772.8	0.135	0.136
Starlink	550	3777.5	0.0126	0.0132

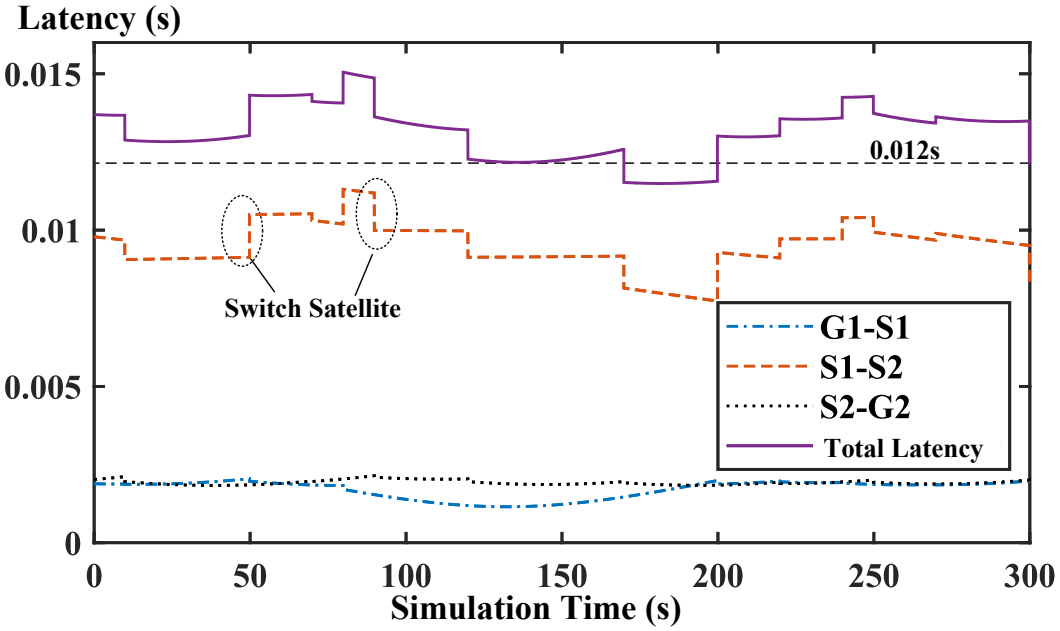

Figure 5.9: LEO data link latency data from 5-minute simulation in RustSat.

based on SDN may be useful to make the connections smoother and more stable via advanced algorithms.

The orbit altitudes and average latencies measured from RustSat are listed in Table 5.3. The data link latencies in RustSat are computed based on physical distances. The SatSDN can simulate the realistic network latency with virtual network interfaces.

The physical power system simulation uses a time-step of 20 μs and the AC-DC systems are partitioned to run on parallel executors based on propagation delays of transmission lines. the MMC-1 and MMC-2 communicate via the SatSDN virtual network and run at real-time speed. When the fault happens,

MMC-1 sends a small fault detection message within 300 bytes, which is small and can be transmitted with low latency. The results of the simulation with different data link latencies are shown in Fig. 5.10.

Fig. 5.10 (a-c) show the DC bus voltage and phase-A voltage waveforms of MMC-2 station with different satellite data paths. The rated voltage level is $\pm 800kV$, resulting in steady-state DC bus voltages of 1600kV. Fig. 5.10 (d-f) show the phase-A voltage waveforms of Bus-8 and Bus-30 in the modified IEEE 118-Bus system. Fig. 5.10 (g-i) show the comparisons of power and current flows with the different satellite data paths.

In the LEO satellite scenario shown in Fig. 5.10 (a), the physical communication latency is a mere 0.0132s, allowing remote messages to be transmitted quicker than fault waveforms. This efficiency origins from vacuum light speed being roughly 33% faster than the speed of light in optical fibers. Therefore, the fault can be cut off before it could impact the power grids in the LEO-based scenario. Fig. 5.10 (d) shows the phase-A voltages at Bus-8 and Bus-30 in the IEEE 118-Bus system, which remain consistent to the steady-state waveforms after $t = 4.0$. Also, for the LEO scenario shown in Fig. 5.10 (g-i), power and currents stay within normal operational ranges without exceeding them.

For the GPS-based and GSO-based scenarios shown in in Fig. 5.10 (b) and (c), the communication latencies are not sufficient to maintain the remote control targets. Although the fault waveform takes approximately 18ms to reach MMC-2, the voltage drops quickly within 0.1s. with GPS-based and GSO-based scenarios; Fig. 5.10 (e) and Fig. 5.10 (f) demonstrate the significant impact on the AC system voltages. As a result, the active power flows shown in Fig. 5.10 (g) and (h) lose control and deviate significantly from their steady-state settings. In Fig. 5.10 (i), the peak AC current levels hit 8kA for GPS-based scenarios and 40kA for GSO-based scenarios.

The case study has shown that the LEO satellite constellation network can act as a reliable backup for power system communications, achieving performance levels unattainable by traditional satellite communication systems. From the results in Fig. 5.10, it can be concluded that the maximum latency for isolating this fault is around 50ms, which is only 3-4 cycles after the fault

happened even when the fault occurs at a remote location. Among the current satellite networks, only LEO satellite constellation network can provide lower latencies compared to traditional wired communication to meet the real-time protection requirements.

5.5 Summary

An ECS-based data-oriented solution composed of RustSat, SatSDN, and ECS-Grid is proposed to realize a real-time cyber-physical digital twin for a wide-area AC-DC grid based on LEO satellite constellation network. The RustSat and ECS-Grid leverage Bevy's ECS framework in Rust, promoting cache-efficient operations and emphasizing the natural alignment between the data-centric characteristic of digital twins and data-oriented design. This approach offers both flexibility and scalability, surpassing traditional object-oriented designs in managing physical and virtual data flows. The ECS foundation enables the functionalities of RustSat and ECS-Grid to be implemented as plugins, further enhancing the system's modularity and extensibility. Additionally, the seamless integration between virtual IEDs in ECS-Grid and MiniNet's virtual network gateways underscores the platform's dedication to authenticity in co-simulation scenarios.

To further improve the proposed digital twin, further accurate approximations of satellite shells and interconnection can be obtained by analyzing the orbital elements in greater detail. More detailed satellite and data link models can be added as a plugin and more SDN intelligent routing algorithms can be used. It is hoped that the insights and data-oriented solution proposed in this work can provide inspiration and assistance to researchers who are interested in exploring the opportunities and challenges of digital-twin technologies and LEO satellite constellation networks in power systems or other industrial applications.

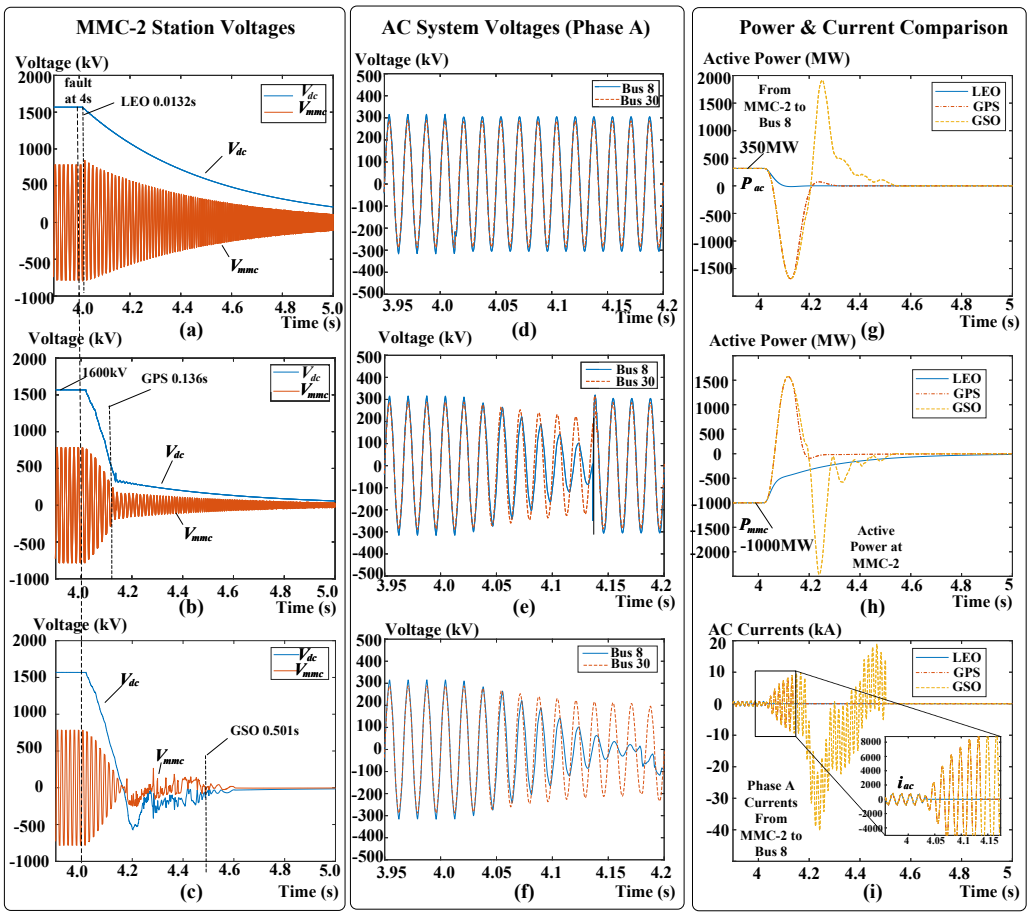

Figure 5.10: Comparative analysis of satellite network latencies on the power grid: (a-c) The DC and AC Phase-A voltages at MMC-2 for Low Earth Orbit (LEO), Global Positioning System (GPS), and Geostationary Orbit (GSO) networks, respectively. (d-f) The Phase-A voltages at Buses 30 and 8 in the modified IEEE 118-Bus system, categorized by the same satellite networks as above. (g) Active power flows at Bus-8 measured in scenarios with LEO, GPS, and GSO satellite networks; (h) Active power flows at MMC-2 measured in scenarios with LEO, GPS, and GSO satellite networks; (i) Phase-A current at Bus-8 measured in scenarios with LEO, GPS, and GSO satellite networks.

Chapter 6

Machine-Learning-Reinforced Massively Parallel Transient Simulation for Large-Scale Renewable-Energy-Integrated Power Systems

6.1 Introduction

[1] This chapter is organized as following: Section 6.2 introduces the fundamentals of machine-learning methodologies and training processes for renewable energy systems (RESs) models. Section 6.3 introduces the heterogeneous CPU-GPU implementation of massively parallel machine-learning RES models based on data-oriented ECS-Grid platform. Section 6.4 shows the test case and results, while the analysis of accuracy and performance is also presented.

RESs are pivotal in the transition to eco-friendly smart grids. Yet, the inherent complexity and uncertainty of these systems, arising from the unpredictability of natural forces such as sunlight and wind, present significant challenges in power system control and operation [154]. Detailed electromagnetic transient (EMT) simulation plays an important role in the analysis of control and operation for RES integrating power systems [155], [156]. However,

[1]This work is accepted: **T. Cheng**, N. Lin and V. Dinavahi, "Machine-learning-reinforced massively parallel transient simulation for large-scale renewable-energy-integrated power systems," *IEEE Trans. Power Syst.*, pp. 1-12, 2024, doi: 10.1109/TPWRS.2024.3409729.

there are more than 300,000 PV panels in a 100MW solar power farm [49], while each module may have an impact on the entire solar farm performance in partial shading scenarios [50], [51], [157]. The same problem also exists for battery groups where the battery management system needs to take care of inconsistencies within the series battery array to maintain the optimal performance [52]. The traditional approach of detailed EMT simulations to address these challenges faces scalability issues due to the computational burden of modeling extensive RES components. For example, the nonlinearity of the PV model requires the Newton-Raphson method to assemble a huge global Jacobian matrix in each iteration, adding prohibitive computational complexity for large-scale power systems with many PV arrays.

A common solution is to utilize massively parallel hardware: Graphical Processing Unit (GPU) to solve large groups of RES components concurrently [66], [67]. However, the nonlinearity of these models limited the solution methods and parallel efficiency as GPUs are not good at complex logics such as branch predictions; nonlinear methods such as the Newton-Raphson method are iterative and needs frequent data exchange between host and device memory, which brings significant overheads. Furthermore, it is challenging to adapt and reimplement complex RES EMT models to highly efficient and scalable GPU codes, making it difficult to keep pace with rapidly evolving new energy and power storage technologies.

Therefore, this chapter proposed to utilize ANN technologies to increase the efficiency of EMT simulation of large-scale systems with RES. ANN is a machine learning technology that can be conceptualized as a mathematical approach to multivariate nonlinear regression [56], which is suitable to reflect the nonlinear RES behaviors including . Given the recent breakthroughs and significant successes of artificial intelligence and machine learning in various complex tasks such as image synthesis [53], natural language processing [54] and weather forecasting [55], these technologies have attracted significant attention from power system researcher as well [57]–[59]. While machine learning technologies are popular in long-term and steady-state power system analysis [60]–[62], the utilization of machine learning technologies is just the begin-

ning for power system EMT simulation. Some research works such as [63]–[65] have demonstrated the advantages and benefits of using ANN and RNN technologies to accelerate real-time EMT models on FPGA. However, these works are early explorations aimed to deal with traditional components in power systems for specific scenarios, and such research hasn't been extended to RES modeling. Moreover, the integration of the ANN models into conventional EMT solvers is important for practical large-scale simulation applications but it was not comprehensively explained in previous research works.

To apply machine learning ANN techniques in accelerating the simulation of large-scale renewable energy models and to extend the previous machine learning EMT model research towards a broader and more practical vision, this work primarily elucidates two key propositions:

- The development and training strategies for neural network modeling of renewable energy generation and energy storage systems. Nonlinear time-variant components are modeled with the gated-recurrent unit (GRU) and time-invariant components such as PV arrays are modeled with the feed-forward network which is also called multi-layer perceptron (MLP). The main focus is on modeling a PV array with multiple independent solar irradiance input variables since solar farms contain a lot of solar panels and each input irradiance may cause significant performance differences under partial shading scenarios.

- this work employs data-oriented entity-component-system (ECS) architecture and GPU instancing strategies to incorporate the ANN model of RES into the EMT power grid simulation program, achieving a highly pragmatic and scalable CPU-GPU massively parallel computing solution. The importance of efficient implementation and integration of machine learning models is often underestimated. This oversight may obstruct the full realization of the models' potential impact on both theoretical advancements and practical applications. Compared to previous research, the proposed design not only establishes a complex multivariate ANN model for photovoltaic panel arrays but also elevates the applica-

tion of the model to a practical level. The integration of ANN models via ECS and plugin-based architectures enables seamless substitution for traditional RES models.

The model training was based on reliable model data produced by traditional physical EMT models and the results were validated with MATLAB/Simulink. The RES components are grouped into a microgrid connected to a synthetic AC/DC system based on the IEEE 118-Bus system, achieving an acceleration performance of 400 times faster than traditional CPU parallel nonlinear iterative computations with more than 2 million RES entities.

6.2 Renewable Energy Systems Neural-Network-Based Modeling

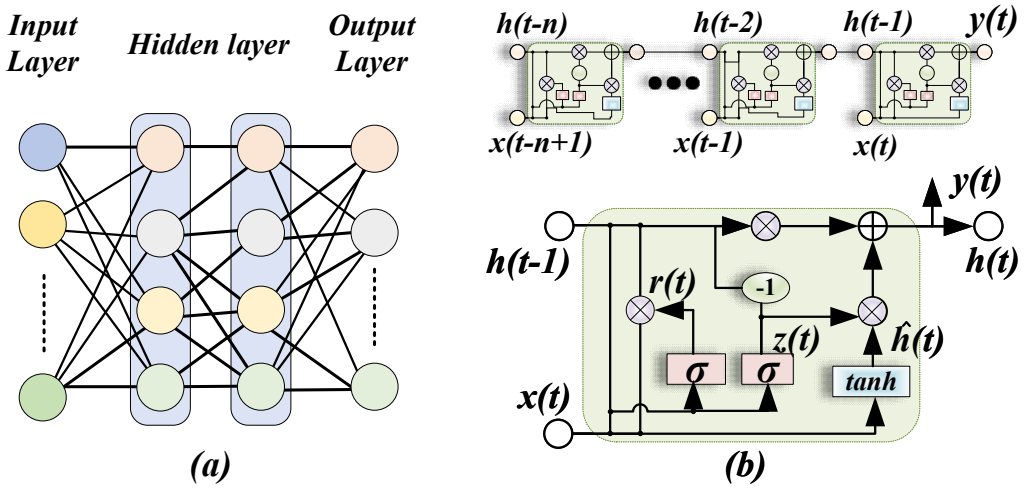

Figure 6.1: Neural network structures: (a) MLP neural network structure; (b) GRU neural network structure.

Traditional EMT models of RESs are mostly nonlinear and require Newton-Raphson's nonlinear iterative method to solve a nonlinear differential-algebraic equation system. Such nonlinear characteristics not only bring heavy computational burden and convergence problems but also limit the efficiency of parallel computing. Machine-learning techniques such as neural networks can effectively capture the nonlinearity of the physics in RESs and provide an accurate approximation to reduce the complexity. It is a fully data-driven process

without many human decisions. Moreover, compared to hand-crafted nonlinear or linearized equivalent models, which have various internal structures and complex computational processes, models trained using neural networks present a consistent matrix computation structure. This uniformity is particularly suited for GPU parallel computations. Thus, machine-learning reinforced RES models can efficiently leverage optimized GPU-accelerated linear algebra libraries such as CUBLAS and CUDNN, without concern for their varying types or internal structures. In addition, the ANN models use *Float32* numbers which is faster than *Float64* required by the Newton-Raphson method on GPUs. The following subsections introduced the basic concept of MLP and GRU neural networks used to model RESs and the training strategies of PV arrays, doubly-fed induction generator (DFIG) wind farms, and batteries.

6.2.1 Multi-Layer Perceptron

The MLP is one of the simplest forms of feed-forward ANNs. It serves as a foundational technique in modern neural network machine learning, yet is sufficiently powerful to address many real-world nonlinear fitting problems. As shown in Fig. 6.1 (a), it contains one input layer, multiple hidden layers, and one output layer. The state variables between hidden layers are connected by activation functions which must be nonlinear functions such as *tanh*, *sigmod*, or *ReLU*. MLP can be expressed by

$$\boldsymbol{z} = f(W\boldsymbol{x} + \boldsymbol{b}) \tag{6.1}$$

where $\boldsymbol{x}$ is the input state vector, $\boldsymbol{b}$ is the bias vector, W is the weight matrix, f is an activation function and $\boldsymbol{z}$ is the output vector of each layer. Although the structure is simple, it is enough to approximate many nonlinear functions of EMT models. Training an MLP model is to solve an optimization problem defined by

$$\min_{W,b} L(\boldsymbol{y}, \hat{\boldsymbol{y}}) \tag{6.2}$$

where $\boldsymbol{y}$ is the output from training or validation dataset, $\hat{y}$ is the prediction data from trained model, and L is a loss function to compute the error between

$\hat{y}$ and y, which can be a mean squared error (MSE) function as the following:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{y}_i - \widehat{\boldsymbol{y}}_i\|^2 \tag{6.3}$$

where n is the total number of samples in the dataset and $||\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i||$ is the L2 normalization for $\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i$.

Gradient descent is often used to solve this minimization problem, which is given by

$$W_{\text{new}} = W_{\text{old}} - \alpha \nabla_W L \tag{6.4}$$

$$\boldsymbol{b}_{\text{new}} = \boldsymbol{b}_{\text{old}} - \alpha \nabla_b L, \tag{6.5}$$

where the gradients $\nabla_W L$ and $\nabla_{\boldsymbol{b}} L$ are computed by back-propagation, and α is a factor called learning rate which controls how large the *old* value change in the gradient direction. The principle of this MLP training process is also valid for other types of neural networks.

6.2.2 Gated Recurrent Unit

GRU networks are based on MLP but have a more complex and specific structure for time-series inputs and outputs. As shown in Fig. 6.1 (b), the typical GRU is expressed by

$$\boldsymbol{z}_t = \sigma(W_{xz}\boldsymbol{x}_t + U_{hz}\boldsymbol{h}_{t-1}) \tag{6.6}$$

$$\boldsymbol{r}_t = \sigma(W_{xr}\boldsymbol{x}_t + U_{hr}\boldsymbol{h}_{t-1}) \tag{6.7}$$

$$\tilde{\boldsymbol{h}}_t = tanh(W\boldsymbol{x}_t + U(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})) \tag{6.8}$$

$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t \tag{6.9}$$

where $\boldsymbol{z}_t$ is the update gate output; $\boldsymbol{r}_t$ is the reset gate output; $\tilde{\boldsymbol{h}}_t$ is the output candidate hidden; $\boldsymbol{h}_t$ is the GRU unit output; W_{xz}, W_{xr}, W are the input weights which are considered; U_{hz}, U_{hr}, U are the recurrent weights and σ is the sigmoid function. The update gate controls the degree to which the hidden state from the previous time step, $\boldsymbol{h}_{t-1}$, should be updated with the new candidate hidden state, $\tilde{\boldsymbol{h}}_t$. It is computed using the sigmoid function, which scales the output between 0 and 1. The reset gate is responsible for

determining the amount of information from the previous hidden state, $\boldsymbol{h}_{t-1}$, that should be retained when computing the output candidate, $\tilde{\boldsymbol{h}}_t$. Similar to the update gate, it also employs the sigmoid function. The output candidate represents a new hidden state based on the input $\boldsymbol{x}_t$ and the previous hidden state $\boldsymbol{h}_{t-1}$. The reset gate, $\boldsymbol{r}_t$, is used to control the influence of $\boldsymbol{h}_{t-1}$ on the output candidate. The output candidate is computed using the hyperbolic tangent function. The final output, $\boldsymbol{h}_t$, is computed by combining the previous hidden state, $\boldsymbol{h}_{t-1}$, and the output candidate, $\tilde{\boldsymbol{h}}_t$, with the help of the update gate, $\boldsymbol{z}_t$. GRU networks are suitable for stateful time-variant components and can reflect more complex behaviors with the price of additional computing steps.

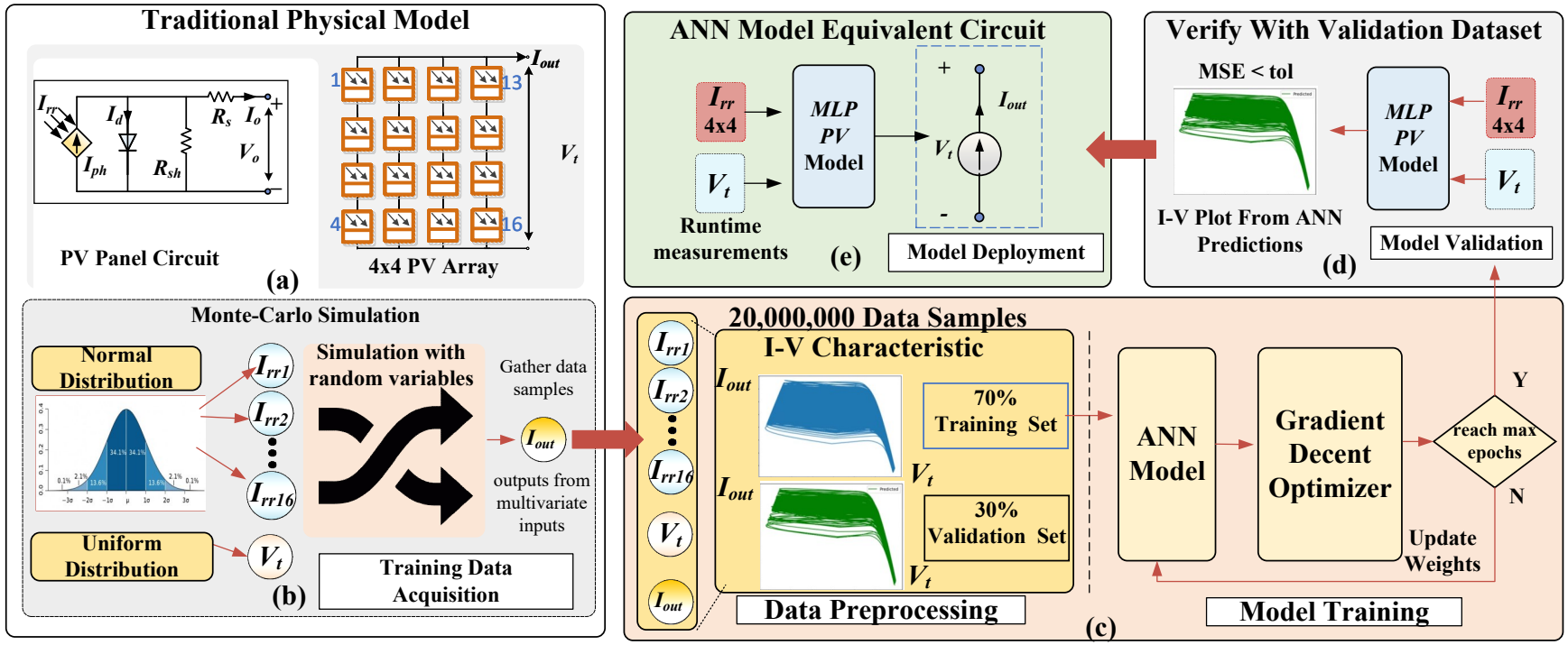

Figure 6.2: MLP modeling of a 4x4 PV array: (a) traditional model of nonlinear PV panel and the 4x4 PV array circuit; (b) Monte Carlo test setup for generating training data; (c) the data preprocessing and training processes of ANN machine learning; (d) model verification on validation dataset; (e) final PV model deploy as a voltage-controlled current source in EMT simulation.

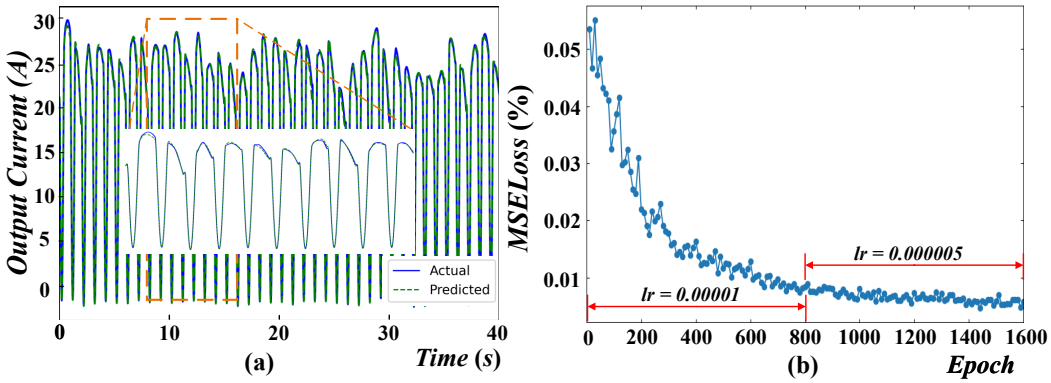

Figure 6.3: PV MLP model training results: (a) output current validation results on the validation data set; (b) MSE loss vs. training epochs.

6.2.3 Machine-Learning MLP Modeling for Photovoltaic Array

As shown in Fig. 6.2 (a), a PV array is constituted by a series-parallel connection of multiple photovoltaic panels. The major electrical characteristic of a PV array is its $I - V$ characteristic under different conditions.

The solar array has 16 independent solar irradiances I_{rr} for each PV panel and a port voltage input V_t. The output is chosen to I_{out} so that the model can be represented by a current source in EMT simulations. Due to the lack of detailed experimental data of a solar array, the training data are produced by traditional EMT simulation of the nonlinear PV model based on the methods and model parameters in [66]. Although the data are derived from simulated circuits, the accuracy of the physical model was verified with real-world experimental data [158] and can produce ideal data for training.

As shown in Fig. 6.2 (b), the Monte Carlo method is applied to effectively cover the state space of the PV array. The irradiance data for each PV array is generated randomly from a normal distribution with a mean of $1000W/m^2$ and a standard deviation of $300W/m^2$. Meanwhile, the port voltage is uniformly sampled from zero to the maximum operational voltage. Using uniform distributions is ideal for training, but normal distributions for solar irradiance better reflect real data and help explore imbalanced dataset impacts on machine learning. This results in the training dataset shown in Fig. 6.2 (c), which covers the full range of operational voltage and a wide range of $I - V$

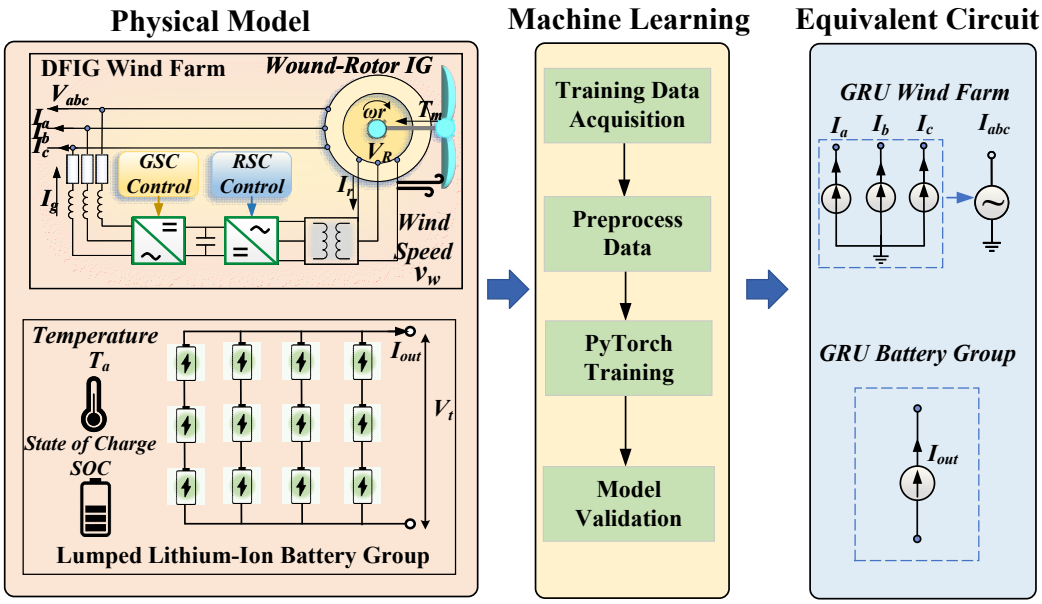

Figure 6.4: GRU-based equivalent circuit models for DFIG wind farms and Lithium-ion battery groups.

characteristics under various irradiance combinations.

A training dataset with 20 million samples was obtained from the testing network. Because the model depends on many independent inputs, it is vital to use dynamic learning rates to achieve good fitting results. Meanwhile, the dropout strategy is used to improve the model's generalization capability and with a dropout rate of 0.25, the error on the validation set was reduced by approximately 20% after using the dropout strategy.

After verifying the accuracy of the trained model with the validation dataset as shown in Fig. 6.2 (d), the model can be deployed into the EMT simulation program to represent the original PV array. As shown in Fig. 6.2 (e), the PV array is represented by a controlled current source for EMT circuit simulation. The output current is predicted by the MLP network comprising four hidden layers, and each layer has 64 cells.

With the proposed training data setup and machine-learning strategies, the trained MLP model yields satisfactory results as shown in Fig. 6.3 (a) and (b). The MSE of the training dataset is about 1e-5 while on the validation dataset, the MSE is around 7e-5, proving the feasibility of the data-driven machine-learning-reinforced modeling for large-scale RESs.

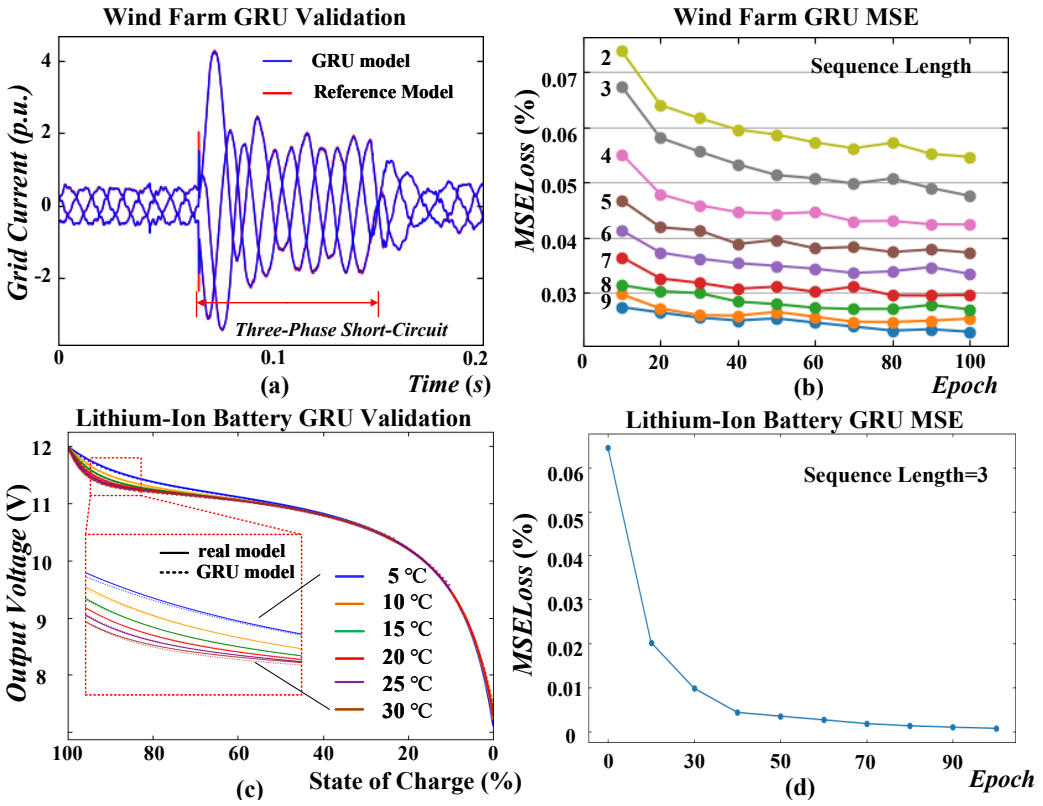

Figure 6.5: Model Results: (a) Phase-A current triggered by a three-phase-to-ground short circuit in the validation dataset. (b) Wind farm GRU Model MSE loss vs. epochs for varied input lengths. (c) Discharge curve validation of Lithium-ion battery GRU model. (d) Battery GRU Model MSE loss vs. epochs.

6.2.4 Machine-Learning GRU Modeling for Wind Farm and Energy Storage

The GRU models were trained using data from a DFIG wind farm and a Lithium-ion battery group simulation as shown in Fig. 6.4. The wind farm test circuit is from [159] and the battery is based on the model used in [67]. The models only use a single layer of GRUs to avoid error accumulations between neural networks. The general machine-learning procedure is similar to the PV array model.

The GRU models use uniformly distributed input variables for convenience. The major difference for GRU is that the GRU uses a sequence of time-series data as inputs so the continuous nature of input signals cannot be violated when generating the data. In this case, each set of parameters should produce

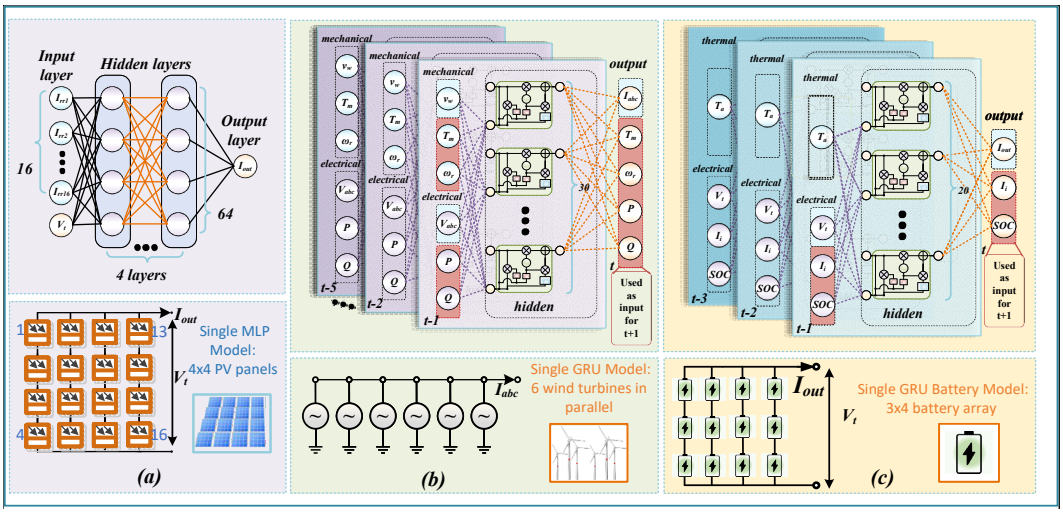

Figure 6.6: Proposed machine-learning-reinforced RES models parameters: (a) MLP model for PV arrays;. (b) GRU model for DFIG wind farms; (c) GRU model for Lithium-ion battery groups.

a contiguous data series within a single Monte Carlo test execution, rather than adjusting parameters mid-run. To simulate fault scenarios, the faulty waveforms can be generated from pre-defined user cases. In this model training, an external three-phase-to-ground fault is added to the datasets used for training the wind farm's GRU model, which takes up 5% of the total data samples.

Fig. 6.5 shows the MSE and validation results of the wind farm and Lithium-ion battery GRU models. Due to the uniform distributed datasets and the complexity of GRU, the training process is much shorter. It takes only 100 epochs to obtain accurate results for both models. Fig. 6.5 (a) shows the comparison between current waveforms of a three-phase-to-ground short circuit fault, where the short circuit resistance is 0.01Ω and the fault duration is 60 milliseconds. The short-circuit fault was applied at the grid connection port of the wind farm. It shows that the GRU model for wind farms successfully captured the fault event even though the fault waveform rarely appears in the dataset. The wind farm model needs a longer GRU input sequence to obtain an accurate GRU model as shown in Fig. 6.5 (b). This is mainly caused by the coupled and time-varied three-phase AC electrical inputs.

The inputs and outputs for the wind farm and battery group model are shown in Fig. 6.4 and Fig. 6.6 (b) and (c).

6.3 Data-Oriented CPU-GPU Heterogeneous Parallel Simulation for Machine-Learning RES Models

The model parameters of proposed machine-learning-reinforced RES models for PV arrays, wind farms, and battery groups are summarized in Fig. 6.6. The following subsections will introduce a data-oriented approach for integrating these ANN models into power system transient simulations, aiming to achieve high performance through CPU-GPU heterogeneous acceleration.

Based on the equations in Section 2.2, the power system EMT simulation program and solver are built using the Rust language and the Bevy ECS framework [160]. The adoption of the ECS architecture for EMT simulation was first proposed in [134]. ECS serves as a data-oriented architecture, emphasizing the efficient layout, storage, and retrieval of data.

Within the data-oriented ECS framework, data and methods are distinctly separate, reflecting the procedural style of C programming. However, this design choice is not a step backward; rather, it provides unparalleled flexibility and performance, making it a preferred approach in modern C-style GPU programming.

As shown in Fig. 6.7 (a), in the Bevy ECS framework, each electrical component and renewable energy source is abstracted as a unique entity, described by a set of distinct data components. For example, a "resistor" entity includes components like resistance value, admittance for circuit matrix solving, and circuit position. In contrast, a "capacitor" entity would have additional components like an equivalent current source, besides its capacitance value, equivalent admittance, and circuit position. Importantly, to unify the treatment in the solver for both three-phase and single-phase circuit elements, an entity with a three-phase topology component will automatically spawn three single-phase child entities to generate three-phase resistors, capacitors, inductors, and

power sources. For some coupled three-phase components such as transformers and transmission lines, there is a special computing system that performs specialized computations on their three-phase components, while they still spawn single-phase child entities for interfacing with the solver. This avoids the need for special three-phase treatments, thereby maintaining solver consistency for single-line DC and three-phase AC power systems.

As shown in Fig. 6.7 (b), the simulation loop consists of three stages, which is the same compared to traditional simulation tools such as PSCAD/EMTDC or other circuit simulators. However, the three stages are generic containers for systems and they serve as schedulers that can generate directed acyclic graphs for parallel executions of systems. All systems are grouped into plugins and users can choose the plugins they need at the runtime. This architecture highlights the ECS framework's flexibility, promoting maximal data components and system reuse. Consequently, this makes integrating new algorithms and models more adaptable and efficient. The entity composition and the flexible plugin features in Bevy ECS will play a significant role in realizing massively parallel computing elegantly in the later subsections.

6.3.1 ANN Model Integrations

Fig. 6.8 shows the entities of RES ANN building blocks for EMT simulation. Renewable sources like PV arrays also contain common components such as equivalent conductance and topological data. However, the computation of their equivalent current sources is determined by specialized components. For entities with conventional nonlinear model parameters, standard algorithms are auto-invoked to compute the PV output. Conversely, entities with artificial neural network model parameters are processed via a specialized GPU compute system, facilitated by a specific neural network plugin: GPUBatchPlugin.

Although the model inference is the same process for all ANN-modeled RES, inputs, and outputs cannot be the same due to different physical structures. Therefore, the preprocessing and postprocessing systems are implemented in specific wind, solar, and battery plugins for convenience. In these model entities, essential environmental inputs like solar irradiance, wind speed,

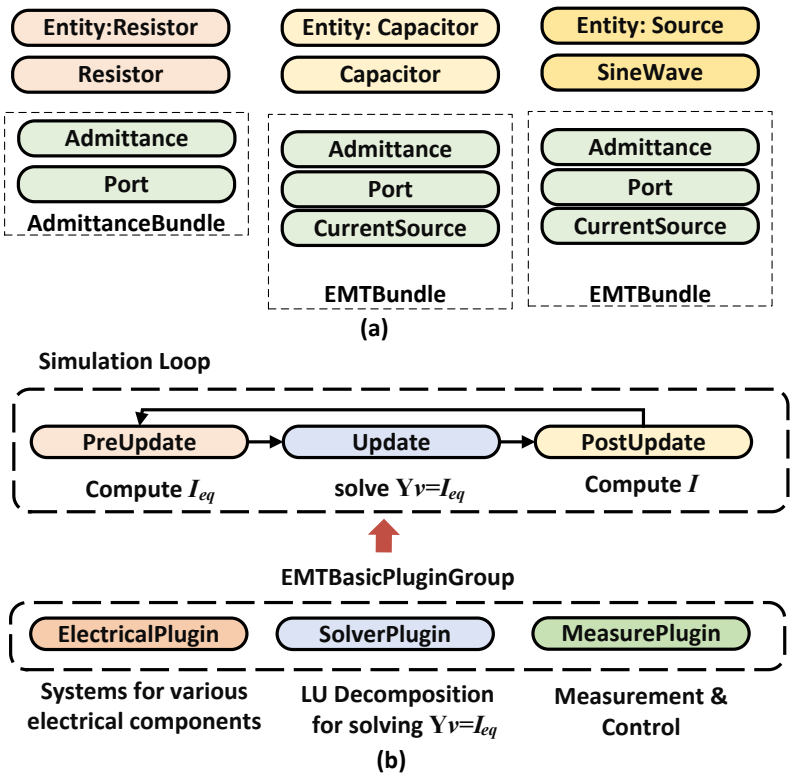

Figure 6.7: (a) Entities of basic circuit components. (b) Basic simulation loop and plugins for proposed data-oriented EMT simulation program.

and ambient temperatures are managed by discrete components. These components are capable of interfacing with user-defined systems to acquire the relevant environmental data. For example, real weather data from a specific region can be integrated into a plugin that can generate these data components without caring about any detail in EMT simulation. Consequently, our proposed design ensures a smooth integration of geographical and weather information with electrical engineering simulation data.

6.3.2 Heterogeneous Massively Parallel CPU-GPU Computing

The ECS framework stores homogeneous data components in cache-friendly contiguous arrays, making it optimal for parallel computing. However, a challenge arises since ECS component data reside on the CPU, and the GPU has its distinct memory management. Efficient data exchange between CPU and GPU

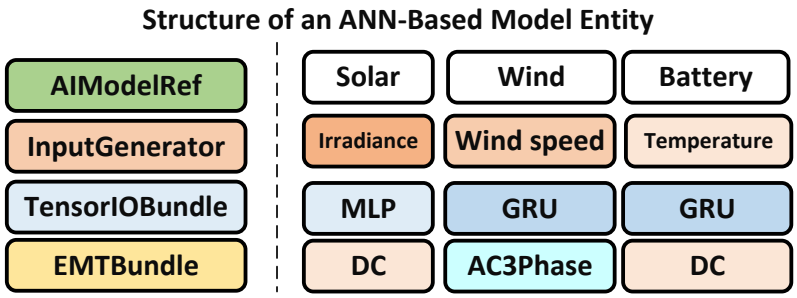

Figure 6.8: The entity variants for PV array, wind farm, and battery groups.

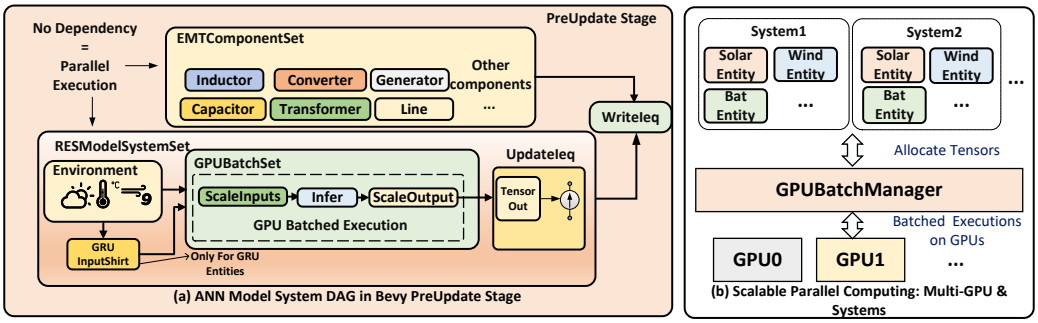

Figure 6.9: Implementation of *GPUBatchPlugin*: (a) Bevy system directed acyclic graph of ANN RES model computation in *PreUpdate* stage; (b) Parallel computing configuration for scaling to multiple circuits and GPUs.

is pivotal for achieving optimal performance. Executing ANN model units individually is not feasible in practice. To address this, a common strategy known as GPU instancing is employed, substantially enhancing performance. This concept of GPU instancing originates from graphical engine development. Initially, it was designed to render multiple 3D objects with identical mesh data in a single batched GPU call, such as rendering a forest composed of the same trees but with varied scales and positions. Analogously, in ANN models, the tree meshes are analogous to matrix weights, and the variations in scale and position correspond to different inputs, outputs, and scalar factors.

A *GPUBatchManager* is built as a singleton in this context, which manages all ANN model and IO tensor memories. At the initial stage, all ANN models will be scanned and registered in *GPUBatchManager*; then, Entities with the same ModelRef are grouped and allocate a global contiguous tensor memory space in *GPUBatchManager* to fit the inputs and outputs; all *TensorIO* components register their data to a specific memory address in these contiguous

GPU memory space according to the type of *ModelRef*. Notice that all tensors must use CPU memory to communicate efficiently with the CPU EMT solver.

As shown in Fig. 6.9 (a), in the *PreUpdate* stage, the CPU will process all ANN inputs and write input data to *TensorIO* which is a local memory pointer mapped to global tensor in *GPUBatchManager*. For GRU, due to the input being a time series, there is an additional input shift system set to store necessary data in the input tensors. Then, instead of processing TensorIO components on the entities one-by-one, the *GPUBatchManager* singleton will copy the global tensor to GPU and perform GPU scaling and inference by model type. Therefore, no matter how many ANN model entities are there, there is only one inter-device copy for all input and output tensors of each model type, which is the reason why tensors should be on host CPU memory to avoid expensive inter-device memory copy overhead. With the managed global tensors, batched model computation is achieved for each model type. Even when dealing with a few hundred ANN model entities, this approach provides a speed-up of more than 100 times compared to executing them individually.

As shown in Fig. 6.9 (b), it's worth noting that the *GPUBatchManager* possesses the capability to associate models with multiple GPUs or streams. Furthermore, the *GPUBatchPlugin* can have an upgraded version to use an upper-level *GPUBatchManager* constructed above multiple ECS objects to manage multiple EMT simulation systems, further amplifying the scalability for GPU instancing. Meanwhile, the intrinsic parallel DAG scheduler of Bevy ECS allows the GPU computations for the ANN models to be executed in parallel with other traditional EMT components whenever feasible. With the power of the ECS framework and GPU instancing techniques, an elegant, adaptable, and highly scalable approach emerges for machine-learning enhanced heterogeneous CPU-GPU massively parallel EMT simulation.

6.4 Study Case and Results

The study case is based on a synthetic AC-DC network based on the IEEE 118-Bus and CIGRE B4 DCS-1 MMC systems, with a RES microgrid connected

at Bus-44 as shown in Fig. 6.10 (a). As shown in Fig. 6.10 (b), the microgrid contains 100 PV farms, one wind farm, and one battery energy storage station. Each PV farm contains 400 MLP-modelled 4x4 PV arrays and is $6400 \times 100 = 640,000$ PV panels in total. The microgrid is connected at Bus-13 with the 25kV/138kV transformer, which is typical in the transmission grid.

As shown in Fig. 6.10 (c), due to the highly optimized EMT simulation program, the computing of AC/DC components, matrix equations solving, and thread synchronization processes only take 20% of overall computation time. The systems in *GPUBatchSet* take a minimal 300μs regardless of the neural network type. This is not only due to heavy computing loads but also caused by much higher overhead to call GPU drivers and move data between CPU and GPU. Therefore, with fine-grained parallel scheduling, the final computing performance is determined by this GPU ANN computing process. This is why tensor components must be on CPU and batch computing must be applied to minimize the overhead.

To extend the scale and complexity of the simulation study case, four IEEE-118 systems are connected together with MMCs as shown in Fig. 6.10. This extends the scale to 2,560,000 machine-learning modeled PV panels and can demonstrate the full power of CPU-GPU massively parallel computing performance under the proposed data-oriented architecture. However, this is only used to extend the system scale and evaluate the parallel performance and the test results will be focused on the RES-related scenarios.

As shown in Fig. 6.10 (d), 8 CPU threads and 2 GPUs are allocated to execute the test system cluster, which can be handled solely by a computing node in the ComputeCanada Cedar cluster. Due to the advanced data-oriented design, this heterogeneous complex computational resource allocation and scheduling are achieved without difficulty.

The test scenarios are mainly related to RES which is a partial shading scenario for PV farms and a wind speed change scenario for wind farms. The results will focus on RES model performance and related system voltage or current changes.

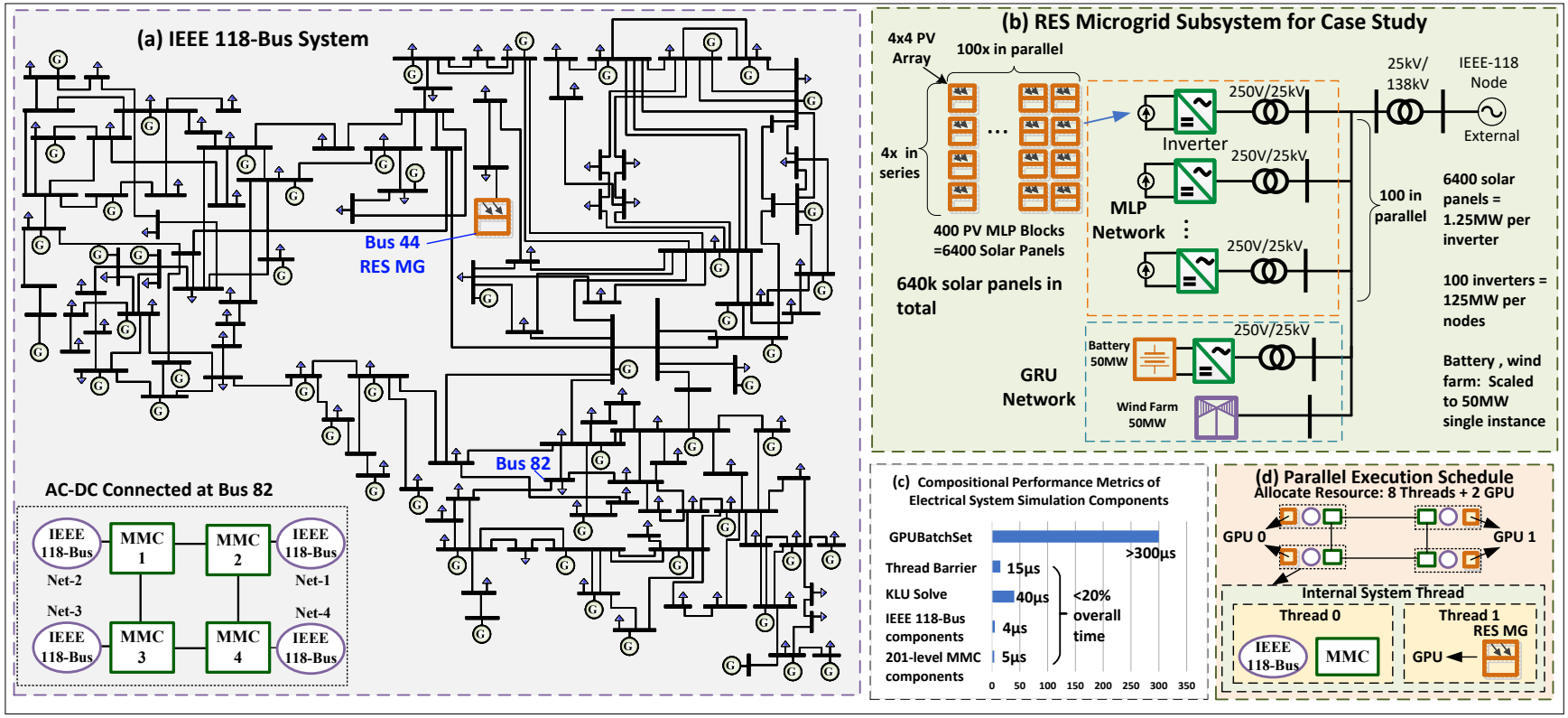

Figure 6.10: Test system: synthetic IEEE 118-Bus-based large-scale AC-DC networks with machine-learning RES building blocks.

6.4.1 Scenario 1: Partial Shading

This scenario sets the rated irradiance of $1000W/m^2$ for all solar panels at the beginning of the simulation. At the $1s$ of simulation time, irradiance of solar panels S1 and S16 to $100W/m^2$ and $200W/m^2$, respectively. Due to the 4x4 PV arrays being serial-connected and S1 and S16 being in separated columns, the power output should be approximately 50% of the rated power due to the output of serial-connected solar panels being determined by the panel with the lowest output current.

Results are shown in Fig. 6.11 (a)-(c), which are measured from the inverter AC side of one PV farm. The PV array achieved very high accuracy with rated irradiance inputs, which only has a relative error of 0.2%. Under the partial shading scenario, the active power of the MLP PV array model drops from 1.25MW to 0.69MW, which has a 4% relative error compared to the original model output. This reflects the impacts of the normally distributed imbalanced dataset which contains 68.27% data samples with solar irradiance of $1000 \pm 300W/m^2$. Despite the increased errors in the peripheral regions, the output waveform shows no significant deviation and still aligns with the characteristics of the photovoltaic model. This, therefore, demonstrates the generalization ability of the MLP model. It's important to keep this in mind for machine learning in the real world because real-world data are often more imbalanced. Using common training methods and MSE loss function may lead to lower errors in areas with more data, but higher errors in areas with less data.

6.4.2 Scenario 2: Wind Speed Step Change

This scenario sets the rated wind speed of $15m/s$ for a wind farm at the beginning of the simulation. At the $5s$ of simulation time, the wind speed changes from $15m/s$ to $10m/s$. The results, as shown in Fig. 6.11 (d)-(f), all exhibit low relative errors, each below 1%. This accuracy mainly comes from the wide range of data used to train the model. The more complex GRU structure also helps to increase the model accuracy for lumped DFIG wind

farm systems. Although the GRU models do have better accuracy for time-series data, the current GRU models use fixed time steps due to the nature of the GRU structure, which limits its usage compared to the MLP PV array model.

The performance is measured from a node in the Cedar cluster of ComputeCanada, which has two NVIDIA® Tesla V100 GPUs. Each Tesla V100 GPU has 5120 CUDA units.

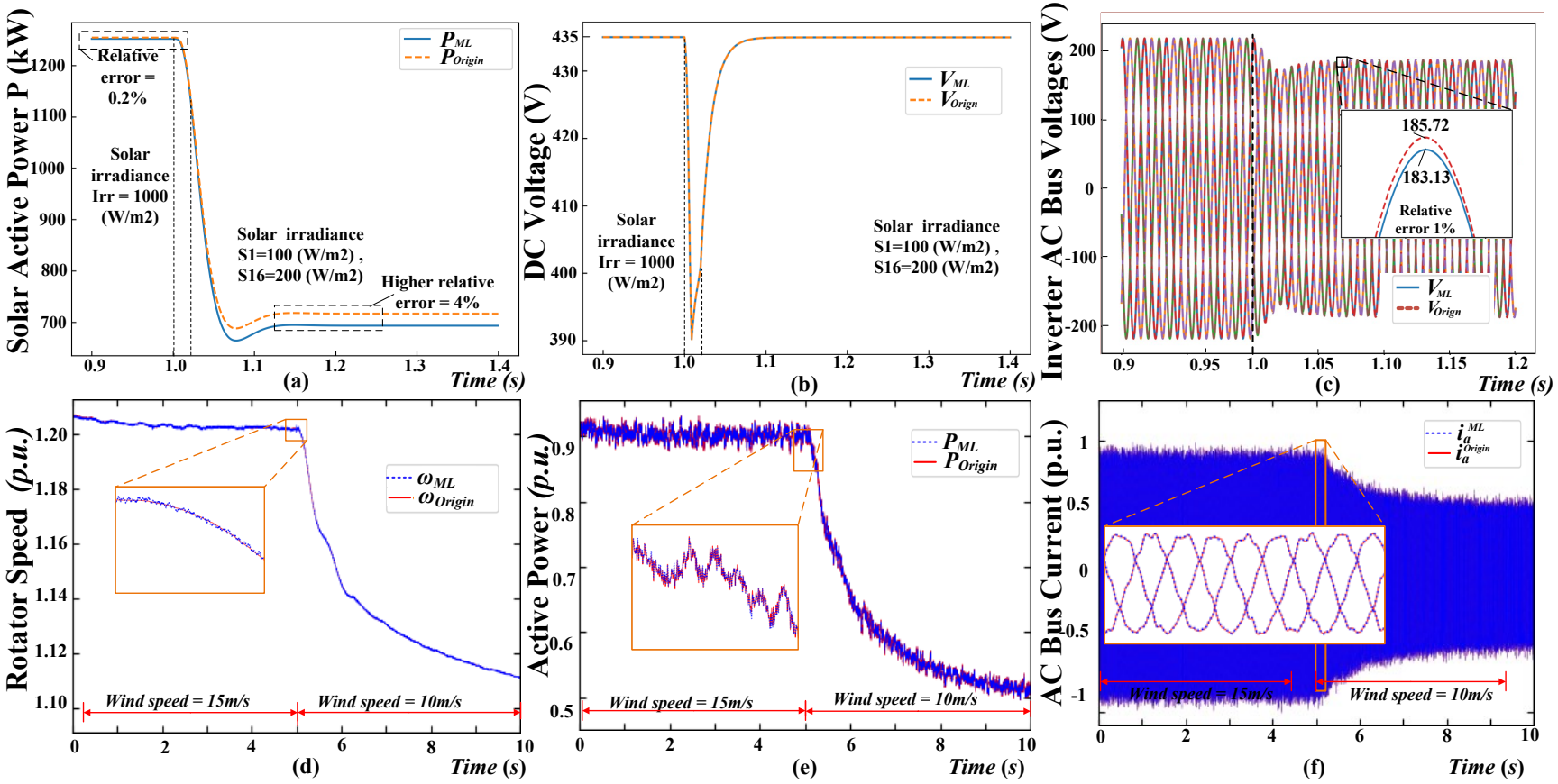

Figure 6.11: Simulation results of test scenarios: Scenario 1: (a) Active power output comparison between MLP and original PV model; (b) DC bus voltage of PV array inverter; (c) AC bus voltages of inverter. Scenario 2: (d) rotor speed comparison between GRU and original DFIG wind farm model; (e) Active power output. (f) Phase-A AC bus current output.

6.4.3 Performance Evaluation

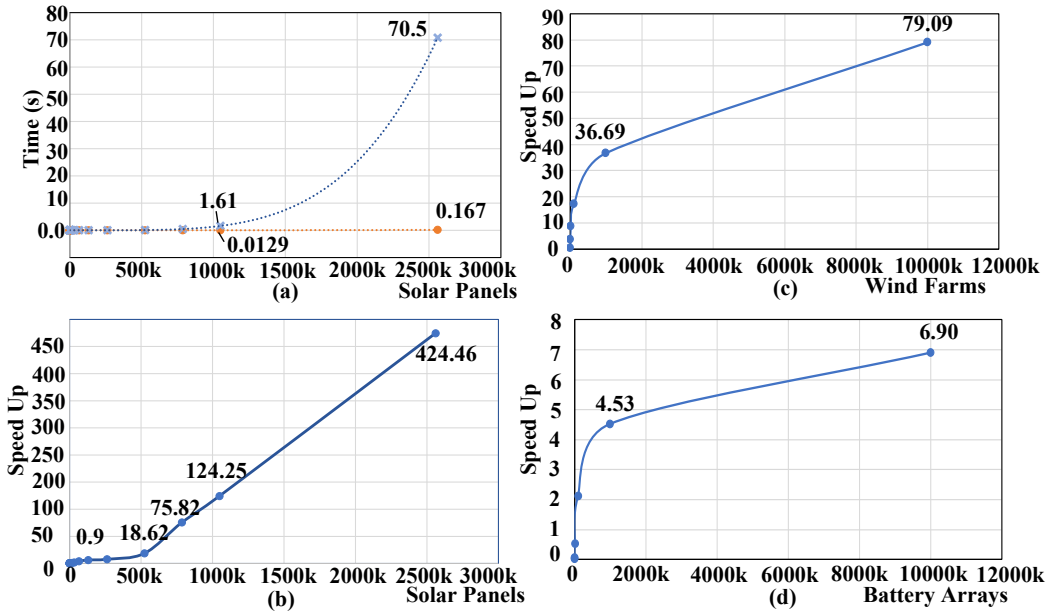

Figure 6.12: (a) Execution time per simulation step vs. number of PV panels of traditional serial CPU nonlinear model and GPU accelerated model; (b) GPU speedup vs. number of PV panels; (c) Wind farm GRU model GPU speed up vs. number of wind farms; (d) Battery array GRU model GPU speed up vs. number of battery arrays.

The performance evaluation of the MLP PV model is performed on the test synthetic system. In Fig. 6.12 (a) and (b), the performance metrics between the CPU-based Newton-Raphson PV array simulation and the GPU-accelerated machine learning alternative are compared. A spectrum of solar panel counts was evaluated to show the distinction between the traditional serial computing approach and the massively parallel GPU-based methodology. The conventional CPU-bound implementation retains its advantage up to the 16k PV panel threshold. Beyond this point, the GPU-facilitated solution demonstrates a consistently low execution time, outperforming the CPU-based Newton-Raphson approach by an order of magnitude. Notably, a speed up exceeding 100 is observed when the system scales beyond 1000k PV panels, equating to approximately 62.5k MLP PV array entities.

For the extensive simulation involving 2560k PV panels, the serial CPU computation paradigm failed to deliver results within an acceptable timescale.

In contrast, the machine learning-driven GPU methodology sustained its efficiency, culminating in an impressive 400x speed-up. This test was accomplished using a dual-GPU setup, as a single GPU is inadequate for managing such an expansive system.

In the evaluated test system, the limited quantity of wind farms and energy storage obscured the potential benefits of GPU acceleration. The performance of GRU models was assessed independently, as shown in Fig.6.12 (c) and (d). Despite the inherent nonlinearity in the physical models of wind farms and batteries, they were implemented via decoupled, non-iterative approaches, resulting in a proportional increase in speed up. However, it is noteworthy that the GRU models required approximately two to three times the execution time compared to MLPs of equivalent system scale. This disparity led to a relatively lower speed-up in Fig.6.12 (c) and (d).

6.5 Summary

This work explores machine-learning-based ANN models for RES components such as PV arrays, DFIG wind farms, and Lithium-ion battery groups, highlighting the significant advancements in machine-learning research and their applications in power system EMT simulations. A data-oriented, heterogeneous CPU-GPU ECS architecture is proposed to realize flexible and fast massively parallel processing of these RES models in large-scale AC/DC power grid simulation. The proposed method has shown promising results for large-scale simulation, achieving high computational accuracy, decent GPU performance, and scalability across various system sizes.

Despite its promising results, there is considerable scope for enhancement in the model's complexity and computational optimization. To derive machine-learning-reinforced digital-twin models for real-world RES stations, training data can be swapped with real-world measurements, which requires collaboration with power generation enterprises for comprehensive on-site research. Moreover, training the ANN models with numerous independent input variables is time-consuming and may not fully cover the operational range. This

can be improved with state aggregation techniques, dictionary learning, and other advanced feature extraction methods. Future work will focus on incorporating real-world measurements and enhancing the efficiency of large-scale RES model training with more sophisticated machine learning technologies.

The potential for practical application of this research is substantial. The RES machine learning models and the data-oriented architecture are not only applicable for EMT simulations but also useful for transient stability simulations. It is hoped that this work will serve as a foundation for future studies, continuing to push the boundaries of power system simulation methods.

Chapter 7

Conclusion

Modern smart grids represent an evolution in the energy sector, driven by advancements in intelligence, clean energy adoption, and comprehensive informatization. As these systems become increasingly complex, the demands on system simulations have expanded dramatically, creating new challenges and opportunities in the realms of transient analysis. In this context, parallel computing has emerged as a crucial tool to enhance simulation performance, offering a means to handle the growing computational demands effectively.

Traditional PiS algorithms have typically been confined to spatial partitioning tasks. However, recent developments in PiT algorithms present promising new methodologies that could significantly extend the capabilities of parallel computing in the analysis of smart grid simulations. This shift points to a broader application of parallel computing technologies, potentially revolutionizing how simulations are conducted and analyzed in the context of modern energy systems. In addition, new data-oriented ECS software architectures and AI technologies are also providing more opportunities for exploring PiS and PiT approaches for power system transient simulation.

7.1 Contributions of Thesis

This thesis explores new methods of parallel computing in modern smart grids. It introduces a simulation framework for AC EMT power systems based on the Parareal algorithm and a hybrid PiT+PiS simulation platform for DC transmission and power electronic converters. This platform leverages an asyn-

chronous heterogeneous CPU-GPU architecture to achieve large-scale transient stability and EMT co-simulation, significantly enhancing simulation speed and efficiency.

Additionally, a data-oriented ECS framework, ECS-Grid, has been developed for simulating cyber-physical power systems. It implements a virtual IED model at the component level, and interfaces with various industrial communication protocols through its plugin and ZeroMQ generic design, supporting large-scale real-time simulation.

To explore new PiS approaches for accelerating nonlinear simulation, a machine learning modeling methodology is proposed to convert traditional nonlinear PV, wind farm and battery models into ANN-based models. This approach reduces computational costs while maintaining input characteristics and is seamlessly integrated into ECS-Grid, utilizing GPU instancing for optimized batch processing execution. The new data-oriented programming paradigm has been proven effective to integrate heterogeneous computing resource, paving the way for a more efficient PiT+PiS algorithm design for the future.

In summary, this study achieves significant results in addressing the challenges of smart grid system simulation, particularly in spatial and temporal parallel computing. It also showcases the broad development prospects of heterogeneous computing hardware structures, data-oriented design philosophies, and data-driven AI technologies in the field of smart grid transient simulation analysis research.The promising data-oriented PiT+PiS AI-enhanced development path has been enlightened for demanding interdisciplinary transient simulations of future smart grids.

7.2 Directions for Future Work

The following research directions are proposed for future works:

Comprehensive ECS Solution for Transient Analysis: The previous research works of ECS-Grid were focused on modeling and establish virtual

IEDs for CPPS simulation. The further research works can explore full ECS implementation for EMT components. In addition, the variable-time-stepping and nonlinear solvers can be implemented as plugins. The same methodology should work with both EMT simulation and TS simulation, while different components and systems should be developed for each simulation type. Ultimately, the ECS-Grid should become an unified platform for all kinds of power transient analysis with heterogeneous GPU and FPGA accelerator supports.

Upgraded Data-Oriented PiT+PiS Algorithm: The PiT+PiS algorithm, which needs multiple coarse-grid and fine-grid workers to perform system-wide time-grid iterations, bring huge burden to implementation for practical applications. With the new data-oriented design, it is now possible to perform vectorized global state update operations directly from archetype tables and eliminate abstract interfaces as well as data copying, which would gain a 30-50% performance improvement. In addition, the PiT+PiS implementation may be compatible with existing plugins for traditional EMT simulation. The data-oriented architecture is promising to achieve a more efficient, flexible and practical PiT+PiS solution for transient simulation of smart grids.

Machine-Learning Modeling and NPU Acceleration for Transient Simulation: The AI technologies bring new data-driven approaches and massively parallel PiS acceleration methods for EMT and TS simulation. The previous research works verified the feasibility of modeling RESs with machine-learning ANN technologies and proposed the integration method between ANN models and data-oriented EMT simulation programs. However, it is just a beginning for a new era of AI applications in smart grid simulation. The future work regards to machine-learning modeling can be focused on modelling more detailed nonlinear RES components, such as internal physics of batteries which is significant for designing battery management systems. Moreover, many advanced neural networks such as physical-informed neural network and auto-encoder may be used to improve the efficiency and accuracy of machine-learning-reinforced EMT models. At last, AI processors have be-

come increasingly prevalent in modern computing environments. Therefore, there is a compelling need to propose new designs and applications aimed at exploring the potential of NPUs in accelerating smart grid simulations, building upon previous advancements.

References

[1] G. Guo, H. Wang, Q. Song, *et al.*, "HB and FB MMC based onshore converter in series-connected offshore wind farm," *IEEE Trans. Power Electron.*, vol. 35, no. 3, pp. 2646–2658, Mar. 2020. 1

[2] N. Lin and V. Dinavahi, "Exact nonlinear micromodeling for fine-grained parallel emt simulation of mtdc grid interaction with wind farm," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 8, pp. 6427–6436, 2019. 2

[3] N. Lin and V. Dinavahi, "Parallel high-fidelity electromagnetic transient simulation of large-scale multi-terminal dc grids," *IEEE Power and Energy Tech. Syst. J.*, vol. 6, no. 1, pp. 59–70, 2019. 2, 36

[4] G. Lauss and K. Strunz, "Multirate partitioning interface for enhanced stability of power hardware-in-the-loop real-time simulation," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 595–605, Jan. 2019. 2, 3

[5] Q. Wang, X. Zhang, Y. Zhang, and Q. Yi, "Augem: Automatically generate high performance dense linear algebra kernels on x86 cpus," in *SC '13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2013, pp. 1–12. 3

[6] M. La Scala, M. Brucoli, F. Torelli, and M. Trovato, "A gauss-jacobi-block-newton method for parallel transient stability analysis (of power systems)," *IEEE Trans. Power Syst.*, vol. 5, no. 4, pp. 1168–1177, Nov. 1990. 3, 51

[7] V. Dolean, P. Jolivet, and F. Nataf, *An introduction to domain decomposition methods: algorithms, theory, and parallel implementation.* SIAM, 2015. 3

[8] S. Zuo, Z. Lin, D. G. Doñoro, X. Zhao, and Y. Zhang, "A massively parallel preconditioned FEM-BEM method for accurate analysis of complex electromagnetic field problems," *IEEE Antennas Wireless Propag. Lett.*, vol. 22, no. 5, pp. 1194–1198, May 2023. 3

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

[10] V. Dinavahi and N. Lin, *Real-Time Electromagnetic Transient Simulation of AC-DC Networks.* Wiley-IEEE Press, 2021. 3

[11] Y. Chen and V. Dinavahi, "Multi-fpga digital hardware design for detailed large-scale real-time electromagnetic transient simulation of power systems," *IET Generation, Transmission & Distribution*, vol. 7, no. 5, pp. 451–463, May 2013. 3

[12] Z. Shen and V. Dinavahi, "Real-time device-level transient electrothermal model for modular multilevel converter on FPGA," *IEEE Trans. Power Electron.*, vol. 31, no. 9, pp. 6155–6168, 2016. 3

[13] T. Liang and V. Dinavahi, "Real-time device-level simulation of MMC-based MVDC traction power system on MPSoC," *IEEE Trans. Transp. Electrific.*, vol. 4, no. 2, pp. 626–641, Jun. 2018. 3

[14] A. Semlyen and F. de Leon, "Computation of electromagnetic transients using dual or multiple time steps," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1274–1281, 1993. 3

[15] N. Lin, P. Liu, and V. Dinavahi, "Component-level thermo-electromagnetic nonlinear transient finite element modeling of solid-state transformer for dc grid studies," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 2, pp. 938–948, 2021. 3

[16] D. Shu, X. Xie, Q. Jiang, G. Guo, and K. Wang, "A multirate EMT co-simulation of large AC and MMC-based MTDC systems," *IEEE Transactions on Power Systems*, vol. 33, no. 2, pp. 1252–1263, 2018. 3

[17] N. Lin, S. Cao, and V. Dinavahi, "Adaptive heterogeneous transient analysis of wind farm integrated comprehensive AC/DC grids," *IEEE Transactions on Energy Conversion*, pp. 1–1, 2020. 3

[18] T. Carraro, M. Geiger, S. Rorkel, and R. Rannacher, *Multiple Shooting and Time Domain Decomposition Methods*. Springer, 2015. 4, 51

[19] G. Gurrala, A. Dimitrovski, S. Pannala, S. Simunovic, and M. Starke, "Parareal in time for fast power system dynamic simulations," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 1820–1830, May 2016. 4, 5, 51, 52

[20] M. Lecouvez, R. D. Falgout, C. S. Woodward, and P. Top, "A parallel multigrid reduction in time method for power systems," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, 2016, pp. 1–5. 4

[21] S. Schöps, I. Niyonzima, and M. Clemens, "Parallel-in-time simulation of eddy current problems using parareal," *IEEE Trans. Magn.*, vol. 54, no. 3, pp. 1–4, Mar. 2018. 4, 51

[22] S. Xia, S. Bu, J. Hu, B. Hong, Z. Guo, and D. Zhang, "Efficient transient stability analysis of electrical power system based on a spatially paralleled hybrid approach," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1460–1473, 2019. 5, 51

[23] V. Jalili-Marandi and V. Dinavahi, "SIMD-based large-scale transient stability simulation on the graphics processing unit," *IEEE Trans. Power Syst.*, vol. 25, no. 3, pp. 1589–1599, 2010. 5, 51, 52, 54, 58

[24] V. Jalili-Marandi, Z. Zhou, and V. Dinavahi, "Large-scale transient stability simulation of electrical power systems on parallel gpus," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1255–1266, 2012. 5, 51, 58

[25] N. Lin, S. Cao, and V. Dinavahi, "Comprehensive modeling of large photovoltaic systems for heterogeneous parallel transient simulation of integrated ac/dc grid," *IEEE Transactions on Energy Conversion*, vol. 35, no. 2, pp. 917–927, Jun. 2020. 5, 51, 63

[26] N. Lin, S. Cao, and V. Dinavahi, "Adaptive heterogeneous transient analysis of wind farm integrated comprehensive ac/dc grids," *IEEE Transactions on Energy Conversion*, vol. 36, no. 3, pp. 2370–2379, 2021. 5, 51

[27] V. Dinavahi and N. Lin, *Parallel Dynamic and Transient Simulation of Large-scale Power Systems*. Springer Nature, 2022. 5, 51

[28] M. La Scala, R. Sbrizzai, and F. Torelli, "A pipelined-in-time parallel algorithm for transient stability analysis (power systems)," *IEEE Trans. Power Syst.*, vol. 6, no. 2, pp. 715–722, May 1991. 5, 51

[29] M. La Scala, G. Sblendorio, and R. Sbrizzai, "Parallel-in-time implementation of transient stability simulations on a transputer network," *IEEE Trans. Power Syst.*, vol. 9, no. 2, pp. 1117–1125, May 1994. 5, 51

[30] B. Park, K. Sun, A. Dimitrovski, Y. Liu, and S. Simunovic, "Examination of semi-analytical solution methods in the coarse operator of parareal algorithm for power system simulation," *IEEE Trans. Power Syst.*, pp. 1–1, 2021. 5, 51

[31] D. Osipov, N. Duan, S. Allu, S. Simunovic, A. Dimitrovski, and K. Sun, "Distributed parareal in time with adaptive coarse solver for large scale power system simulations," *2019 IEEE Power Energy Society General Meeting (PESGM)*, pp. 1–5, 2019. 5, 52

[32] *PSCAD User's Guide v4.6*, PSCAD, Feb. 2024. [Online]. Available: `https://www.pscad.com/knowledge-base/article/160`. 5

[33] H. W. Dommel, "Digital computer solution of electromagnetic transients in single-and multiphase networks," *IEEE Trans. Power App. Syst.*, vol. PAS-88, no. 4, pp. 388–399, 1969. 5

[34] H. W. Dommel, *EMTP theory book*. Microtran Power System Analysis Corporation, 1996. 5, 15, 23

[35] D. Gutierrez-Rojas, P. H. J. Nardelli, G. Mendes, and P. Popovski, "Review of the state of the art on adaptive protection for microgrids based on communications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 1539–1552, 2021. 6, 72

[36] H. Georg, S. C. Müller, C. Rehtanz, and C. Wietfeld, "Analyzing cyber-physical energy systems:the INSPIRE cosimulation of power and ICT systems using hla," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2364–2373, 2014. 6, 73, 98

[37] H. Lin, S. S. Veda, S. S. Shukla, L. Mili, and J. Thorp, "GECO: Global event-driven co-simulation framework for interconnected power system and communication network," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1444–1456, 2012. 6, 73

[38] M. Thornton, M. Motalleb, H. Smidt, J. Branigan, P. Siano, and R. Ghorbani, "Internet-of-things hardware-in-the-loop simulation architecture for providing frequency regulation with demand response," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 5020–5028, 2018. 6, 73

[39] G. Cao, W. Gu, C. Gu, *et al.*, "Real-time cyber physical system co-simulation testbed for microgrids control," *IET Cyber-Phys. Syst. Theory Appl.*, vol. 4, Feb. 2019. 6, 73

[40] T. Duan and V. Dinavahi, "Starlink space network-enhanced cyber-physical power system," *IEEE Trans. Smart Grid*, vol. 12, no. 4, pp. 3673–3675, 2021. 6

[41] A. A. Jahromi, A. Kemmeugne, D. Kundur, and A. Haddadi, "Cyber-physical attacks targeting communication-assisted protection schemes," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 440–450, Jan. 2020. 6, 73

[42] I. Ali and S. M. Suhail Hussain, "Communication design for energy management automation in microgrid," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 2055–2064, May 2018. 6, 73

[43] *Entt: Gaming meets modern c++*, GitHub, Jul. 2022. [Online]. Available: `https://github.com/skypjack/entt`. 6, 7, 76

[44] *Data-oriented technology stack (DOTS)*, Unity, Jul. 2022. [Online]. Available: `https://unity.com/dots`. 6, 7, 76

[45] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, 2149–2154 vol.3. 7, 106

[46] *About gazebo*, Open Robotics, Feb. 2024. [Online]. Available: `https://gazebosim.org/about`. 7

[47] Y. Yuan, K. Dehghanpour, Z. Wang, and F. Bu, "Multisource data fusion outage location in distribution systems via probabilistic graphical models," *IEEE Trans. Smart Grid*, vol. 13, no. 2, pp. 1357–1371, 2022. 7

[48] BevyEngine. "Bevy: A refreshingly simple data-driven game engine built in Rust free and open source forever!" Accessed on Apr. 8, 2024. (2024), [Online]. Available: `https://bevyengine.org`. 7, 99, 105

[49] UN Climate Change Conference, *100MW solar PV power plant at quaid-e-azam solar park, lal sohanra, cholistan, bahawalpur, pakistan*, Accessed: 2023-11-06, 2018. [Online]. Available: `https://cdm.unfccc.int/Projects/DB/RWTUV1493361547.6/view`. 7, 124

[50] H. Oufettoul, N. Lamdihine, S. Motahhir, N. Lamrini, I. A. Abdelmoula, and G. Aniba, "Comparative performance analysis of pv module positions in a solar pv array under partial shading conditions," *IEEE Access*, vol. 11, pp. 12 176–12 194, 2023. 8, 124

[51] A. Djalab, N. Bessous, M. M. Rezaoui, and I. Merzouk, "Study of the effects of partial shading on PV array," in *2018 International Conference on Communications and Electrical Engineering (ICCEE)*, 2018, pp. 1–5. 8, 124

[52] A. T. Elsayed, C. R. Lashway, and O. A. Mohammed, "Advanced battery management and diagnostic system for smart grid infrastructure," *IEEE Tran. Smart Grid*, vol. 7, no. 2, pp. 897–905, Mar. 2016. 8, 124

[53] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, *High-resolution image synthesis with latent diffusion models*, 2022. arXiv: 2112.10752 [cs.CV]. 8, 124

[54] S. Mohamadi, G. Mujtaba, N. Le, G. Doretto, and D. A. Adjeroh, *ChatGPT in the age of generative AI and large language models: A concise survey*, 2023. arXiv: 2307.04251 [cs.CL]. 8, 124

[55] K. Bi, L. Xie, H. Zhang, *et al.*, "Accurate medium-range global weather forecasting with 3d neural networks," *Nature*, vol. 619, pp. 533–538, 2023. 8, 124

[56] A. Burkov, *The Hundred-page Machine Learning Book*. Andriy Burkov, 2019. 8, 124

[57] F. Li and Y. Du, "From alphago to power system ai: What engineers can learn from solving the most complex board game," *IEEE Power Energy Mag.*, vol. 16, no. 2, pp. 76–84, Mar. 2018. 8, 124

[58] S. R, R. S. Kumar, and A. T. Mathew, "Online static security assessment module using artificial neural networks," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4328–4335, Nov. 2013. 8, 124

[59] B. Huang and J. Wang, "Applications of physics-informed neural networks in power systems - a review," *IEEE Trans. Power Syst.*, vol. 38, no. 1, pp. 572–588, Jan. 2023. 8, 124

[60] H. Long, R. Geng, W. Sheng, H. Hui, R. Li, and W. Gu, "Small-sample solar power interval prediction based on instance-based transfer learning," *IEEE Trans. Ind. Appl.*, vol. 59, no. 5, pp. 5283–5292, 2023. 8, 124

[61] M. Cui, F. Li, H. Cui, S. Bu, and D. Shi, "Data-driven joint voltage stability assessment considering load uncertainty: A variational bayes inference integrated with multi-CNNs," *IEEE Trans. Power Syst.*, vol. 37, no. 3, pp. 1904–1915, May 2022. 8, 124

[62] Z. Yi, Y. Xu, X. Wang, *et al.*, "An improved two-stage deep reinforcement learning approach for regulation service disaggregation in a virtual power plant," *IEEE Trans. Smart Grid*, vol. 13, no. 4, pp. 2844–2858, 2022. 8, 124

[63] S. Zhang, T. Liang, T. Cheng, and V. Dinavahi, "Machine learning based modeling for real-time inferencer-in-the-loop hardware emulation of high-speed rail microgrid," *IEEE Trans. Emerg. Sel. Topics Power Electron.*, vol. 3, no. 4, pp. 920–932, 2022. 8, 125

[64] S. Zhang, T. Liang, and V. Dinavahi, "Hybrid ML-EMT-based digital twin for device-level HIL real-time emulation of ship-board microgrid on FPGA," *IEEE J. Emerg. Sel. Top. Ind. Electron.*, vol. 4, no. 4, pp. 1265–1277, Oct. 2023. 8, 125

[65] S. Cao, V. Dinavahi, and N. Lin, "Machine learning based transient stability emulation and dynamic system equivalencing of large-scale AC-DC grids for faster-than-real-time digital twin," *IEEE Access*, vol. 10, pp. 112 975–112 988, 2022. 8, 125

[66] N. Lin, S. Cao, and V. Dinavahi, "Comprehensive modeling of large photovoltaic systems for heterogeneous parallel transient simulation of integrated AC/DC grid," *IEEE Transactions on Energy Conversion*, vol. 35, no. 2, pp. 917–927, Jun. 2020. 8, 124, 131

[67] N. Lin, S. Cao, and V. Dinavahi, "Massively parallel modeling of battery energy storage systems for AC/DC grid high-performance transient simulation," *IEEE Trans. Power Syst.*, vol. 38, no. 3, pp. 2736–2747, May 2023. 8, 124, 133

[68] M. J. Gander and S. Vandewalle, "Analysis of the parareal time-parallel time-integration method," *SIAM Journal on Scientific Computing*, vol. 29, no. 2, pp. 556–578, 2007. 18

[69] M. J. Gander and E. Hairer, "Nonlinear convergence analysis for the parareal algorithm," in *Domain decomposition methods in science and engineering XVII*, Springer, 2008, pp. 45–56. 20

[70] L. Wang, J. Jatskevich, V. Dinavahi, *et al.*, "Methods of interfacing rotating machine models in transient simulation programs," *IEEE Trans. Power Del.*, vol. 25, no. 2, pp. 891–903, 2010. 22

[71] T. Duan and V. Dinavahi, "Adaptive time-stepping universal line and machine models for real-time and faster-than-real-time hardware emulation," *IEEE Trans. Ind. Electron.*, pp. 1–10, 2019. 22, 23

[72] Y. Kuang, *Delay differential equations: with applications in population dynamics*. Academic press, 1993. 23, 24

[73] M. Günther and U. Feldmann, "The dae-index in electric circuit simulation," *Math. Comput. Simulat.*, vol. 39, no. 5-6, pp. 573–582, 1995. 24

[74] S. Peyghami, P. Davari, M. Fotuhi-Firuzabad, and F. Blaabjerg, "Standard test systems for modern power system analysis: An overview," *IEEE Industrial Electronics Magazine*, vol. 13, no. 4, pp. 86–105, 2019. 27

[75] R. Champagne, L.-A. Dessaint, H. Fortin-Blanchette, and G. Sybille, "Analysis and validation of a real-time ac drive simulator," *IEEE Trans. Power Electron.*, vol. 19, no. 2, pp. 336–345, 2004. 33

[76] H. Saad, S. Dennetière, J. Mahseredjian, *et al.*, "Modular multilevel converter models for electromagnetic transients," *IEEE Trans. Power Del.*, vol. 29, no. 3, pp. 1481–1489, 2014. 33, 34

[77] Q. Tu and Z. Xu, "Impact of sampling frequency on harmonic distortion for modular multilevel converter," *IEEE Trans. Power Del.*, vol. 26, no. 1, pp. 298–306, 2011. 33

[78] P. Kuffel, K. Kent, and G. Irwin, "The implementation and effectiveness of linear interpolation within digital simulation," *International Journal of Electrical Power and Energy Systems*, vol. 19, no. 4, pp. 221–227, 1997. 34, 35

[79] M. Faruque, V. Dinavahi, and W. Xu, "Algorithms for the accounting of multiple switching events in digital simulation of power-electronic systems," *IEEE Transactions on Power Delivery*, vol. 20, no. 2, pp. 1157–1167, Apr. 2005. 35

[80] ABB, *ABB 5SNA-2000K450300 StakPak IGBT module*, Accessed: 2020-06-04, Jun. 2020. [Online]. Available: `https://new.abb.com/products/5SNA2000K450300/stakpak-igbt-module`. 37

[81] H. Bai, C. Liu, A. K. Rathore, D. Paire, and F. Gao, "An FPGA-based IGBT behavioral model with high transient resolution for real-time simulation of power electronic circuits," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6581–6591, Aug. 2019. 38

[82] *The OpenMP API specification for parallel programming*, Accessed: 2020-06-04, Jun. 2020. [Online]. Available: `https://www.openmp.org/`. 39

[83] T. Cheng, T. Duan, and V. Dinavahi, "Parallel-in-time object-oriented electromagnetic transient simulation of power systems," *IEEE Open Access J. Power Energy*, vol. 7, pp. 296–306, 2020. 39

[84] *Changji-Guquan UHVDC transmission project*, 2015. [Online]. Available: `http://www.nsenergybusiness.com/projects/changji-guquan-uhvdc-transmission-project/`.. 51

[85] *Dolwin 5 HVDC project*. [Online]. Available: `https://www.hitachiabb-powergrids.com/ca/en/references/hvdc/dolwin-5`. 51

[86] *Critical grid infrastructure to connect the west*. [Online]. Available: `http://www.transwestexpress.net/`. 51

[87] D. Shu, X. Xie, Z. Yan, V. Dinavahi, and K. Strunz, "A multi-domain co-simulation method for comprehensive shifted-frequency phasor DC-grid models and EMT AC-grid models," *IEEE Trans. Power Electron.*, vol. 34, no. 11, pp. 10 557–10 574, Nov. 2019. 51

[88] Y. Takahashi, K. Fujiwara, T. Iwashita, and H. Nakashima, "Parallel finite-element method based on space-time domain decomposition for magnetic field analysis of electric machines," *IEEE Trans. Magn.*, vol. 55, no. 6, pp. 1–4, Jun. 2019. 51

[89] "Ieee guide for synchronous generator modeling practices and parameter verification with applications in power system stability analyses," *IEEE Std 1110-2019 (Revision of IEEE Std 1110-2002)*, pp. 1–92, 2020. 52

[90] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-Hill New York, 1994, pp. 45–198. 52, 54

[91] "IEEE recommended practice for excitation system models for power system stability studies," *IEEE Std 421.5-2005 (Revision of IEEE Std 421.5-1992)*, pp. 1–93, 2006. 53

[92] V. Vittal, J. D. McCalley, P. M. Anderson, and A. Fouad, *Power system control and stability*. John Wiley & Sons, 2019. 54

[93] W. Janischewskyj and P. Kundur, "Simulation of the non-linear dynamic response of interconnected synchronous machines part i-machine modelling and machine-network interconnection equations," *IEEE Trans. Power App. Syst.*, vol. PAS-91, no. 5, pp. 2064–2069, 1972. 54

[94] B. Stott, "Power system dynamic response calculations," *Proceedings of the IEEE*, vol. 67, no. 2, pp. 219–241, 1979. 56

[95] J. L. Gustafson, "Encyclopedia of parallel computing," in *Encyclopedia of Parallel Computing*, D. Padua, Ed. Boston, MA: Springer US, 2011, pp. 53–60. 56

[96] M. Minion, "A hybrid parareal spectral deferred corrections method," *Commun. Appl. Math. Comput. Sci.*, vol. 5, no. 2, pp. 265–301, 2010. 56

[97] *Thrust: Code at the speed of light*. [Online]. Available: `https://github.com/NVIDIA/thrust`. 60

[98] *Unified memory for cuda beginners*, Jun. 2017. [Online]. Available: `https://developer.nvidia.com/blog/unified-memory-cuda-beginners`. 64

[99] *Maximizing unified memory performance in cuda*, Nov. 2017. [Online]. Available: `https://developer.nvidia.com/blog/maximizing-unified-memory-performance-cuda`. 64

[100] H. J. Schellnhuber, S. Rahmstorf, and R. Winkelmann, "Why the right climate target was agreed in paris," *Nature Climate Change*, vol. 6, no. 7, pp. 649–653, 2016. 72

[101] S. Bolognani, R. Carli, G. Cavraro, and S. Zampieri, "On the need for communication for voltage regulation of power distribution grids," *IEEE Trans. Control. Netw. Syst.*, vol. 6, no. 3, pp. 1111–1123, Sep. 2019. 72

[102] N. Patari, V. Venkataramanan, A. Srivastava, D. K. Molzahn, N. Li, and A. Annaswamy, "Distributed optimization in distribution systems: Use cases, limitations, and research needs," *IEEE Trans. Power Syst.*, [Early Access], 2021. 72

[103] D. K. Molzahn, F. Dörfler, H. Sandberg, *et al.*, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Trans. Smart Grid*, vol. 8, no. 6, pp. 2941–2962, Nov. 2017. 73

[104] R. V. Yohanandhan, R. M. Elavarasan, P. Manoharan, and L. Mihet-Popa, "Cyber-physical power system (CPPS): A review on modeling, simulation, and analysis with cyber security applications," *IEEE Access*, vol. 8, pp. 151 019–151 064, 2020. 73

[105] K. Mets, J. A. Ojea, and C. Develder, "Combining power and communication network simulation for cost-effective smart grid analysis," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1771–1796, 2014. 73

[106] IEEE Task Force on Interfacing Techniques for Simulation Tools, S. C. Müller, H. Georg, *et al.*, "Interfacing power system and ICT simulators: Challenges, state-of-the-art, and case studies," *IEEE Trans. Smart Grid*, vol. 9, no. 1, pp. 14–24, Jan. 2018. 73

[107] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, K. Birman, and D. Coury, "EPOCHS: A platform for agent-based electric power and communication simulation built from commercial off-the-shelf components," *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 548–558, May 2006. 73

[108] M. H. Cintuglu, O. A. Mohammed, K. Akkaya, and A. S. Uluagac, "A survey on smart grid cyber-physical system testbeds," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 446–464, Jan. 2017. 73, 82, 98

[109] E. Moradi-Pari, N. Nasiriani, Y. P. Fallah, P. Famouri, S. Bossart, and K. Dodrill, "Design, modeling, and simulation of on-demand communication mechanisms for cyber-physical energy systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2330–2339, 2014. 73

[110] B. Chen, K. L. Butler-Purry, A. Goulart, and D. Kundur, "Implementing a real-time cyber-physical system test bed in RTDS and OPNET," in *2014 North American Power Symposium (NAPS)*, Nov. 2014, pp. 1–6. 73, 98

[111] M. Ni, H. Tong, L. Zhao, and M. Li, "A flexible hardware-in-the-loop testbed for cyber physical power system simulation," *IET Cyber-Phys. Syst. Theory Appl.*, vol. 4, Jul. 2019. 73

[112] H. Tong, M. Ni, L. Zhao, and M. Li, "A flexible hardware-in-the-loop testbed for cyber physical power system simulation," *IET Cyber-Phys. Syst., Theory Appl.*, vol. 4, Dec. 2019. 73, 98

[113] *An open-source universal messaging library*, The ZeroMQ authors, Jul. 2022. [Online]. Available: `https://zeromq.org/`. 74, 85

[114] *Messagepack*, GitHub, Jul. 2022. [Online]. Available: `https://github.com/msgpack/msgpack`. 74, 84

[115] I. Jahn, F. Hohn, G. Chaffey, and S. Norrga, "An open-source protection ied for research and education in multiterminal hvdc grids," *IEEE Trans. Power Syst.*, vol. 35, no. 4, pp. 2949–2958, 2020. 78

[116] *FlatBuffer serialization C++ benchmarks*, Google FlatBuffer, Nov. 2022. [Online]. Available: `https://google.github.io/flatbuffers/flatbuffers_benchmarks.html`. 84

[117] *Theoretical maximum memory bandwidth for intel® core™ x-series processors*, Intel, Jul. 2021. [Online]. Available: `https://www.intel.com/content/www/us/en/support/articles/000056722/processors/intel-core-processors.html`. 84

[118] *eProsima Fast DDS*, eProsima, Nov. 2022. [Online]. Available: `https://www.eprosima.com/index.php/products-all/eprosima-fast-dds`. 85

[119] *Paho is an IoT project*, Eclipse Foundation, Nov. 2022. [Online]. Available: `https://www.eclipse.org/paho/`. 85

[120] *Eclipse Mosquitto™*, Eclipse Foundation, Nov. 2022. [Online]. Available: `https://mosquitto.org/`. 85

[121] *Industroyer2: Industroyer reloaded*, welivesecurity, Nov. 2022. [Online]. Available: `https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded/`. 87

[122] C. Marnay, C. Abbey, G. Joos, *et al.*, "Microgrids 1 engineering, economics, & experience," *Electra; CIGRE*, 2015. 89

[123] L. Thurner, A. Scheidler, F. Schäfer, *et al.*, "Pandapower — an open-source python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Trans. Power Syst.*, vol. 33, no. 6, pp. 6510–6521, Nov. 2018. 89, 113

[124] Y. Henri, *Handbook of Small Satellites: Technology, Design, Manufacture, Applications, Economics and Regulation*, J. N. Pelton, Ed. Cham: Springer International Publishing, 2020, pp. 1–10. 96

[125] B. Al Homssi, A. Al-Hourani, K. Wang, *et al.*, "Next generation mega satellite networks for access equality: Opportunities, challenges, and performance," *IEEE Commun. Mag.*, vol. 60, no. 4, pp. 18–24, 2022. 96

[126] J. Foust, "SpaceX's space-internet woes: Despite technical glitches, the company plans to launch the first of nearly 12,000 satellites in 2019," *IEEE Spectrum*, vol. 56, no. 1, pp. 50–51, 2019. 96

[127] North American Electric Reliability Corporation (NERC). "Loss of communication to transmission substations." Accessed on Apr. 8, 2024. (Nov. 2023), [Online]. Available: `https://www.nerc.com/pa/rrm/ea/Lessons%20Learned%20Document%20Library/LL20231101_Loss_of_Communication_to_Transmission_Substations.pdf`. 97

[128] Q. Yang, D. I. Laurenson, and J. A. Barria, "On the use of leo satellite constellation for active network management in power distribution networks," *IEEE Trans. Smart Grid*, vol. 3, no. 3, 1371–1381, 2012. 97

[129] K. Sun, W. Qiu, Y. Dong, *et al.*, "Wams-based hvdc damping control for cyber attack defense," *IEEE Trans. Power Syst.*, vol. 38, no. 1, pp. 702–713, 2023. 97

[130] K. E. Holbert, G. Heydt, and H. Ni, "Use of satellite technologies for power system measurements, command, and control," *Proc. IEEE*, vol. 93, no. 5, pp. 947–955, 2005. 97

[131] V. Madani, A. Vaccaro, D. Villacci, and R. L. King, "Satellite based communication network for large scale power system applications," in *2007 iREP Symposium - Bulk Power System Dynamics and Control - VII. Revitalizing Operational Reliability*, 2007, pp. 1–7. 97

[132] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 5–20, 2013. 97

[133] W. B. Group, *Myanmar - development of a myanmar national electrification plan towards universal access 2015-2030 (english)*, `http://documents.worldbank.org`, Washington, D.C., 2015. 97

[134] T. Cheng, T. Duan, and V. Dinavahi, "ECS-Grid: Data-oriented real-time simulation platform for cyber-physical power systems," *IEEE Trans. Ind. Informat.*, vol. 19, no. 11, pp. 11 128–11 138, Nov. 2023. 98, 105, 135

[135] T. Duan, T. Cheng, and V. Dinavahi, "Heterogeneous real-time co-emulation for communication-enabled global control of ac/dc grid integrated with renewable energy," *IEEE Open J. Ind. Electron. Soc.*, vol. 1, pp. 261–270, 2020. 98

[136] D. Bhattacherjee and A. Singla, "Network topology design at 27,000 km/hour," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, ser. CoNEXT '19, Orlando, Florida: Association for Computing Machinery, 2019, pp. 341–354. 98

[137] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019. 99

[138] M. Liu, Y. Gui, J. Li, and H. Lu, "Large-scale small satellite network simulator: Design and evaluation," in *2020 3rd International Conference on Hot Information-Centric Networking (HotICN)*, 2020, pp. 194–199. 99

[139] S. Kassing, D. Bhattacherjee, A. B. Águas, J. E. Saethre, and A. Singla, "Exploring the "internet from space" with hypatia," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20, Virtual Event, USA: Association for Computing Machinery, 2020, pp. 214–229. 99, 118

[140] Q. Chen, J. Guo, L. Yang, X. Liu, and X. Chen, "Topology virtualization and dynamics shielding method for LEO satellite networks," *IEEE Commun. Lett.*, vol. 24, no. 2, pp. 433–437, 2020. 100

[141] W. Jiang, "Software defined satellite networks: A survey," *Dig. Commun. Netw.*, 2023. 100

[142] D. Vallado, *Fundamentals of Astrodynamics and Applications (4th Edition)*. Hawthorne, CA, USA: Microcosm, 2013, pp. 188–189. 100, 102

[143] F. R. Hoots and R. L. Roehrich, "Models for propagation of NORAD element sets," Aerospace Defense Command Peterson AFB CO Office of Astrodynamics, Tech. Rep., 1980. 100

[144] D. Vallado, P. Crawford, R. Hujsak, and T. Kelso, "Revisiting spacetrack report #3," in *2006 AIAA/AAS Astrodynamics Specialist Conference*, vol. 3, Aug. 2006. 101

[145] T. S. Kelso. "CelesTrak: Current NORAD two-line element sets." Accessed on Apr. 8, 2024. (), [Online]. Available: `https://celestrak.org`. 101

[146] A. Marcireau. "The SGP4 algorithm ported to Rust from the reference CelesTrak implementation." Accessed on Apr. 8, 2024. (), [Online]. Available: `https://github.com/neuromorphicsystems/sgp4`. 101

[147] B. R. Bowring, "Transformation from spatial to geographical coordinates," *Survey Review*, vol. 23, no. 181, pp. 323–327, 1976. 103

[148] B. R. Bowring, "The accuracy of geodetic latitude and height equations," *Survey Review*, vol. 28, no. 218, pp. 202–206, 1985. 103

[149] E. Stefanakis, "Web mercator and raster tile maps: Two cornerstones of online map service providers," *Geomatica*, vol. 71, no. 2, pp. 100–109, 2017. 103

[150] Mininet. "An instant virtual network on your laptop." Accessed on Apr. 8, 2024. (), [Online]. Available: `http://mininet.org`. 105, 110

[151] D. Merkel, "Docker: Lightweight linux containers for consistent development and deployment," *Linux J.*, vol. 2014, no. 239, Mar. 2014. 105

[152] S. Yehonathan, *Data-Oriented Programming: Reduce software complexity*. Manning, 2022. 106

[153] B. Rhodes, *Skyfield: High precision research-grade positions for planets and Earth satellites generator*, Astrophysics Source Code Library, record ascl:1907.024, Jul. 2019. ascl: 1907.024. 118

[154] X. Liang, "Emerging power quality challenges due to integration of renewable energy sources," *IEEE Trans. Ind. Appl.*, vol. 53, no. 2, pp. 855–866, Mar. 2017. 123

[155] S.-K. Kim, J.-H. Jeon, C.-H. Cho, E.-S. Kim, and J.-B. Ahn, "Modeling and simulation of a grid-connected PV generation system for electromagnetic transient analysis," *Solar Energy*, vol. 83, no. 5, pp. 664–678, 2009. 123

[156] J. Marchand, A. Shetgaonkar, J. L. Rueda Torres, A. Lekic, and P. Palensky, "EMT real-time simulation model of a 2 GW offshore renewable energy hub integrating electrolysers," *Energies*, vol. 14, no. 24, 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/24/8547. 123

[157] M. Adly and K. Strunz, "Efficient digital control for MPP tracking and output voltage regulation of partially shaded PV modules in DC bus and DC microgrid systems," *IEEE Trans. Power Electron.*, vol. 34, no. 7, pp. 6309–6319, Jul. 2019. 124

[158] M. S. E. Muljadi and V. Gevorgian, "PSCAD modules representing PV generator," *National Renewable Energy Lab. (NREL) Technical Reports*, vol. 35, no. 2, pp. 917–927, Aug. 2013. 131

[159] N. Lin and V. Dinavahi, "Exact nonlinear micromodeling for fine-grained parallel emt simulation of MTDC grid interaction with wind farm," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6427–6436, Aug. 2019. 133

[160] Bevy Engine Contributors. "Bevy: A refreshingly simple data-driven game engine built in Rust." Accessed on: Nov. 20, 2023. (), [Online]. Available: https://bevyengine.org/. 135

Appendix A

Top level code of CPU-GPU asynchronous program

```
len = 10000, coarse_len = 10, fine_len = 100
tol = 1e-4
# objects for coarse and fine-grid workers
coarse_network = create_coarse_network()
fine_networks = create_fine_networks()
# coarse_network is initalized with larger dt

# solution vectors for Parareal
Gk, Gk_1, Fk,Fk_1, Uk = create_parareal_vectors()


# TS-EMT data exchange vector, unified memory
PQ, Vtheta = create_TSEMT_vectors()
iter = 0
maxit = 10
converged = False
Gstream, Fstreams = create_streams()
for i in range(0, len):
  while not converged and iter < maxit:
    # OMP Thread 1: # EMT simulation Task
    Prepare_emt_sim()
    EMT_solve()
    PrefetchDataToGPU(PQ)

    # OMP Thread 2: # TS simulation Task
    # set intial values
    Gk[0] = Uk[0]
    coarse_network = Uk[0]
    fine_networks[0] = Uk[0]
    # 10 intervals need 9 predictions
    for x in range(1, coarse_len):
      # PiS GPU functions are called in
      # the solve function
      TS_coarse_solve(
          coarse_network, Gk[x]
          PQ, Vtheta, Gstream)

      if iter == 0:
        # (a) coarse prediction
        # copy functions are async kernel functions
        # the stream is assigned at runtime
        copy_from_to(Gk[x], Uk[x], Gstream)
      else:
        # (c) solution refinement
        err = Gk[x] - Gk_1[x]
        converged = converged & check_error(err, tol)
        Uk[x] = Fk_1[x * fine_len] + err
        copy_from_to(Uk[x], coarse_network, Gstream)

      copy_from_to(Uk[x], finenet[x], Gstream)
```

```
        # (b) Fine-grid parallel operation
        for xt in range(0, fine_len):
          TS_fine_solve(
              fine_networks[x-1],
              PQ, Vtheta, Fstreams[x-1])
          Fk_idx = x*fine_len+xt+1
          copy_from_to(
              fine_networks[x-1],
              Fk[Fk_idx], Fstreams[x-1])
        # ensure results before next serial operation
        cudaStreamSynchronize(Gstream)

      copy_from_to(Gk, Gk_1, Gstream)

      # device only synchronize once in 1 iteration
      # since fine-grids kernels are concurrent
      cudaDeviceSynchronize()
      copy_from_to(Fk, Fk_1, Gstream)
      PrefetchDataToCPU(Vtheta)
  # (d) the final solution of a window is there
```

Appendix B

ZeroMQ C++ Plugin Header for ECS-Grid

```cpp
#pragma once
// details of components
#include "components.hpp"
#include "entt/entity/fwd.hpp"
// details of systems
#include "vIED/zmq/IEDSystems.hpp"

// ECS container
#include "world/world.hpp"

/// This is your module namespace
namespace vIED
{
namespace zmq
{
/// declare your components
namespace Components
{
struct ZMQIEDContext;
struct IEDMessageBuffer;
struct IED;
struct IEDSocket;

/// a set of IED components
/// can be grouped into a bundle
using IEDBundle = basic::Bundle<IED,
IEDMessageBuffer, IEDSocket, Name>;

}  // namespace Components

/// declare your customized stage
enum class vIEDStage
{
  Communicate

};

/// declare your systems
namespace vIEDSystem
{
/// this initialize context and sockets
void initialize(entt::registry& reg);
/// this send messages using tx socket
/// including serialization
void sender(entt::registry& reg);
/// this receive messages using rx socket
/// including deserialization
void receiver(entt::registry& reg);
};  // namespace vIEDSystem

```

```
51  struct Plugin
52  {
53    static void build(ecs::World& world)
54    {
55      // register your systems to the simulator
56      using namespace vIEDSystem;
57
58      world.add_startup_system_to_stage(PostStartUp,
59                                        initialize);
60      auto&& stage =
61      world.add_stage_after(PostUpdate,
62                            vIEDStage::Communicate);
63
64      world.add_system_to_stage(stage->name(), sender);
65      world.add_system_to_stage(stage->name(), receiver);
66    }
67  };
68  }  // namespace zmq
69  }  // namespace vIED
```