1-1-2006

A Generalized Three-Phase Coupling Method For Distributed Simulation Of Power Systems

Jian Wu

A GENERALIZED THREE-PHASE COUPLING METHOD FOR DISTRIBUTED SIMULATION OF POWER SYSTEMS

By

Jian Wu

A Dissertation
Submitted to the Faculty
of Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Electrical Engineering
in the Department of Electrical and Computer Engineering

Mississippi State, Mississippi

August 2006

GENERALIZED THREE PHASE COUPLING METHOD FOR

DISTRIBUTED SIMULATION OF

POWER SYSTEMS

By

Jian Wu

Approved:

Noel N. Schulz Associate Professor of Electrical and Computer Engineering (Major Advisor and Director of Thesis)	Herbert L. Ginn III Assistant Professor of Electrical and Computer Engineering (Committee Member)
J. W .Bruce Associate Professor of Electrical and Computer Engineering (Committee Member)	Yoginder S. Dandass Assistant Professor of Computer Science Engineering (Committee Member)
James C. Harden Professor and Department Head of Electrical and Computer Engineering	Roger L. King Associate Dean of the College of Engineering

Name: Jian Wu

Date of Degree: August 5, 2006

Institution: Mississippi State University

Major Field: Electrical Engineering

Major Professor: Dr. Noel N. Schulz

Title of Study: A GENERALIZED THREE-PHASE COUPLING METHOD FOR DISTRIBUTED SIMULATION OF POWER SYSTEMS

Pages in Study: 103

Candidate for Degree of Doctor of Philosophy

The simulation of power system behavior is a highly useful tool for planning, analysis of power system stability, and operator training. Traditionally, small power system studies are dominated by the time taken to solve the machine dynamics equations, while larger studies are dominated by the time taken to solve the network equations. With the trend towards more sophisticated and realistic modeling, the size and complexity of simulations of a power system are growing tremendously. The ever-increasing need for computational power can be satisfied by the application of distributed simulation.

Power systems are distributed in nature. The terrestrial power systems are divided into groups and controlled by different Regional Transmission Organization (RTO). Each RTO owns the detailed parameter for the area under control, but only limited data and boundary measurement of the external network. Thus, performing power system analysis in such cases is a challenge. Also, simulating a large-scale power system with detailed modeling of the components creates a heavy computational burden.

One possible way of relieving this problem is to decouple the network into subsystems and solve the subsystem respectively, and then combine the results of the subsystems to get the solution. The way of decoupling a network and representing the missing part will affect the result greatly. Also, due to information distribution in the dispatch centers, the problem of doing power system analysis with limited data available arises. The equivalents for other networks need to be constructed to be able to analyze the power system.

This research work proposes a distributed simulation algorithm to handle the issues above. It introduces the history of distributed simulation, proposes a generalized coupling method dealing with natural coupling, and develops and demonstrats distributed simulation models are in theVirtual Test Bed (VTB). The models undergoes tests with different network configurations. A presentation and analysis of the test results follow. The performance of the distributed simulation compares satisfactorily with the steady state result and time domain simulation result.

DEDICATION

This dissertation is dedicated to my beloved family.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my academic advisor, Dr. Noel Schulz, who has given the direction and support throughout my graduate program for her constant help and support both technically and emotionally during my studies.

I would also like to thank my committee members, Dr. Herbert L. Ginn III, Dr. Yoginder Dandass and Dr. J. W. Bruce for their advice during my research and course work.

I sincerely appreciate Mississippi State University and the Office of Naval Research for their financial support during my thesis work. I also convey thanks to Dr. Wenzhong Gao for his insightful discussions on this research subject.

I will always remember all my lab members here at Mississippi State University for their help and support during my studies and research. I quite enjoyed the time being with them at Mississippi State University.

Special thanks are given to my family members, my daughter, my husband, my parents and my sisters for their encouragement, understanding and support.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

LIST OF ABBREVIATIONS

RTO	Regional Transmission Organization
SPS	Shipboard Power Systems
VTB	Virtual Test Bed
RTM	Remote Testing and Measurement
MURI	Multiple University Research Initiative
DAQ	Data Acquisition Device
LAN	Local Area Network
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
MAC	Medium Access Control
DAE	Differential Algebraic Equations
TLM	Transmission Line Matrix
RCF	Resistive Companion Form
KCL	Kirchhoff's Current Law
RPC	Remote Procedure Call
IDL	Interface Definition Language
CORBA	Common Object Request Broker Architecture
IPC	Inter Process Communication
NDR	Network Data Representation
DLL	Dynamic Link Library
MAPE	Mean Absolute Percentage Error

CHAPTER I

INTRODUCTION

This chapter introducs the distributed simulation of power system applications and the shipboard power system. It presents the objectives and contributions of this dissertation. The dissertation sections are outlined.

1.1 Introduction to the problem

The simulation of power system behavior is a highly useful tool for planning, analysis of stability, and operator training. Traditionally, solving the machine dynamics equations dominates the computation time of small power system studies, while solving the network equations dominates the computationn time of larger system. With the trend towards more sophisticated and realistic modeling, the size and complexity of simulations of a power system are growing tremendously. The ever-increasing need for computational power can be satisfied by the application of distributed simulation. Distributed simulation integrates separately and concurrently, computational sub-units to find the solution of a large-scale power system. Distributed simulation can be applied to both terrestrial power systems and shipboard power systems.

1.1.1 Terrestrial powers systems

The large-scale terrestrial power systems are divided into groups and controlled by different Regional Transmission Organizations (RTOs), which coordinate distinct portions of the power systems and are responsible for keeping the power system reliable and safe. Each RTO has the detailed data for the area under control, but only limited data and boundary measurements for the external network. After the deregulation of the power system, the external network model is not clearly available for the dispatch center. As a result, a tremendous challenge to perform comprehensive power system analysis to this case.

Therefore, distributed simulation, which can perform analysis with only the local and boundary data available for each site, is beneficial to large-scale power system analysis.

Second, with more intelligent devices being introduced, the simulation of a power system with intelligent control is even more difficult. Intelligent control may involve neural networks, optimization, or artificial intelligence.

Co-simulating the control aspects along with the power system analysis challenging within one simulation tool. Furthermore, many of the intelligent devices only provide external connection information, and detailed models are not available.

1.1.2 Shipboard power systems

An all-electric ship differs from a conventional terrestrial power system. For better understanding of the functionality and influence introduced by new hardware to the electric ship, both the equipment and the controllers, testing the new devices essential

before putting them into use. However, designing new ship systems involves constructing full-scale prototypes, which is both costly and hardware intensive. While testing with a real electric ship is costly and risky, a virtual test environment is more affordable and safe for performing hardware tests in the prototype stage. Such type of hardware in loop tests can be taken as distributed simulation with part simulated in software and some of the response originating from hardware.

Therefore, distributed simulation, which can decouple an entire system at the power level and the signal level coupling point to multiple sites, is beneficial to large-scale power system analysis and SPS (Shipboard Power Systems) analysis. Distributed simulation helps quickly diagnose failures in SPS and enables better understanding of the system status. An extension of distributed simulation could enable hardware to interact remotely [1,2].

Recent advances in real-time digital network simulation methods and technologies can help to create a virtual test environment. Distributed simulation can help the analysis of large-scale shipboard power systems which requires for fast diagnosis of failures. Improving testing and analysis of shipboard power systems can enhance failure diagnosis and provide greater insight into the system status. Its extension could also allow multiple hardware test facilities to interact remotely [1,2].

1.2 Objectives

The objective of this research is to develop a computationally efficient and reliable distributed simulation algorithm, based on agent technology.

A power system computation can be divided into sub-networks and solved individually. When the detailed model of some areas is hidden from others, the systems can be divided and solved based on the boundary value response. Therefore, the decomposition allows not only a more simplified problem formulation process due to the small size of a decoupled network, but more importantly, it opens the way to parallel simulation.

The sub-tasks of developing a distributed simulation algorithm are below:

- Development of load models for transient study that conforms steady state analysis. Signal ports use facilitates distributed simulation by making a load controllable from external.
- Implementation and testing of the load models on various power systems.
- Development of a new computationally efficient and reliable distributed simulation algorithm method to enable natural decoupling and signal level decoupling, for use in transient analysis.
- Implementation and testing of the agent based distributed simulation on various power systems.

1.3 Contribution of proposed work

The contribution of this distributed algorithm and load model follows:

- The new developed decoupled model will enable distributed simulation at both the natural coupling level and the signal coupling level, which can hide the model details from each side and has the potential to boost computational solution speed.

- With the proposed tools, these research activities work to integrate control algorithm testing with power system simulation.
- The newly-developed load models enable the user to perform transient analysis for element described by steady state parameter.
- The newly-developed load models enable the user to control the load remotely and monitor the test results over the network.

1.4 Outline of dissertation

The organization of this dissertation follows:

Chapter 2 serves as an introduction to distributed simulation. First, it introduces the project background, which is followed by an illustration of power system modeling and simulation and then a review of existing methods used in distributed simulation.

Chapter 3 summarizes the limitations of existing methods, and justifues the need for an improved distributed simulation algorithm. The technology and simulation tools for this research are briefly introduced. It presents a work plan for developing a new distributed simulation algorithm and load models for transient study as well.

Chapter 4 describes the development of the load model. It develops a conventional polynomial model and then a RLC-based PQ load model to enable time domain simulation. Their mathematical models are derived and implemented in the Virtual Test Bed (VTB). It presents and analyzes their performance in steady state and transient study analysis.

Chapter 5 describes the development of the agent based distributed algorithm. The natural coupling model and signal coupling models are implemented in the VTB. These

models' performance in steady state and transient study are presented and analyzed to verify the correctness and applicability of the proposed algorithm.

Chapter 6 presents the conclusion and future work.

CHAPTER II

BACKGROUND AND LITERATURE REVIEW

This chapter introduces the project background, followed by an illustration of power system modeling and simulation and review of existing methods used in distributed simulation.

2.1 Background

2.1.1 MURI introduction and steps

Motivated by reasons stated in chapter 1, five universities in the United States teamed up to develop a Remote Testing and Measurement (RTM) device and procedures to virtually connect power system laboratories over the internet. Funded by the Department of Defense, the MURI (Multiple University Research Initiative) project works at setting up a large-scale power system laboratory to carry advanced, non-destructive testing and measurement of power systems. A conceptual diagram of remote software to a hardware power system setup is in Figure 2.1 below [1]:

Five desired remote interconnection and experiments are considered:

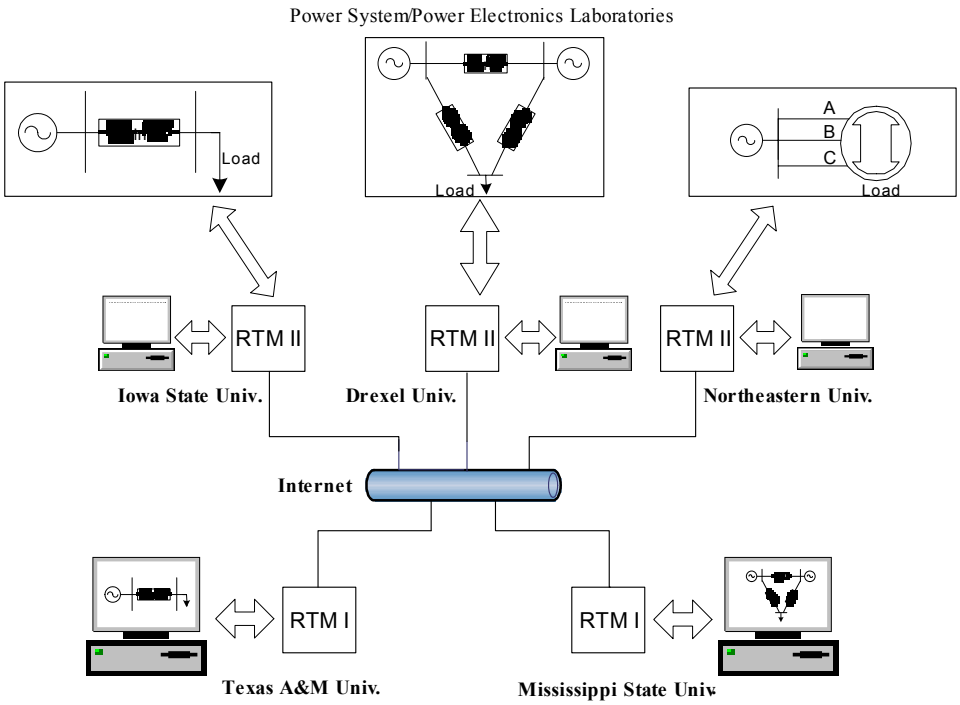


Figure 2.1 Conceptual Remote Testing and Measurement Setup Between Participating Universities [1]

- Implementation I: software power simulation tools to software power simulation tools.
- Implementation II: software power simulation tools to hardware power systems.
- Implementation III: hardware power systems to hardware power systems.
- Implementation IV: software power simulation tools to multiple hardware systems.
- Implementation V: multiple interconnected hardware power systems.

2.1.2 Final Structure of MURI

The final goal of this project is to have remotely distributed power equipment work together. The theory of remote testing and measurement (RTM) is in development. Figure 2.2 shows the main structure of RTM for remote test.

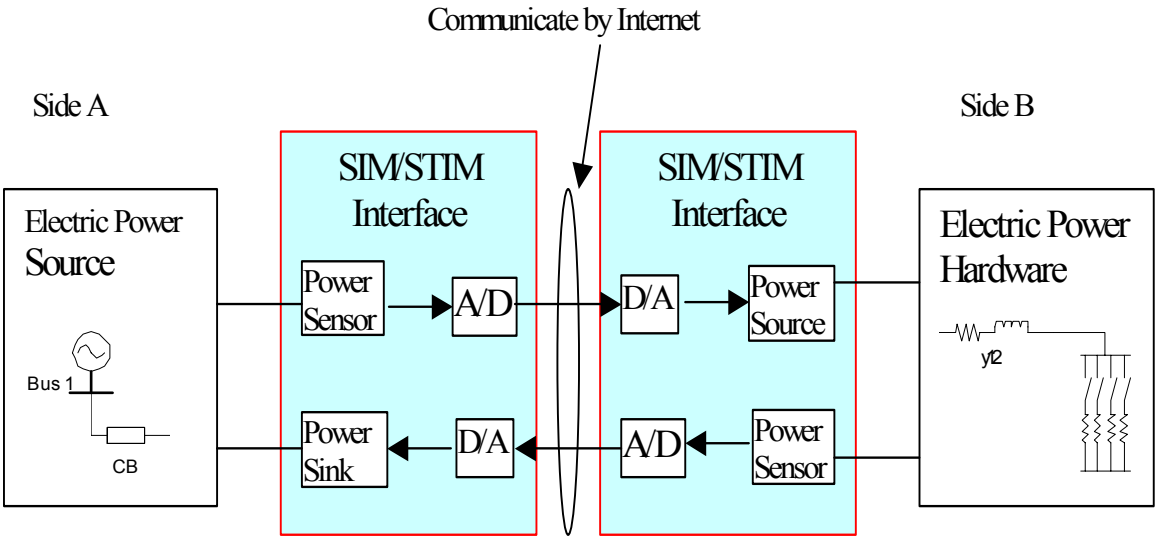


Figure 2.2 Structure of Remote Hardware/Hardware Implementation

In this structure, the hardware of the power source and power load is located at different universities, and they work together virtually from both sides via an internet connection. At each side, there are power sensors and data acquisition devices to collect information about the real-time status of the power source and power load. They exchange the status information continuously through communication ports on both sides. Once status information reaches one side, they construct a corresponding load (sinking current) at side A and power source (supply voltage) at side B. So the hardware at each side is working as if it is really connected to the remote hardware.

For the simple case shown in Figure 2.2, the detailed scenario of the process would be: initially, the power source starts, and the voltage level information transfers from side A to side B through the network. At side B, a signal controlled voltage source connects to the load. At side B, power sensor and data acquisition device (DAQ) acquires the load response (absorbed current) to the power source. Through communication ports, side B transfers the load profile back to side A.

At side A, a phantom load (power sink) could be constructed at the Sim/Stim interface according to the received load profile without knowing details of the load components. Therefore, the generator responds to the load just as if it is connected to the real load. The generator response to the load then could be fed back to the load side.

In this structure, a Simulation-Stimulation (SIM/STIM) interface is used at both sites. This interface must either generate or absorb power, so real power must be exchanged between the simulation and the hardware. The existence of Sim-Stim interface will alter the power hardware's performance relative to its performance when the hardware connects directly. The selection of the Sim-Stim Interface parameters (such as sampling period and time delays) will affect the performance of the system relative to that of the real system performance. Papers [3, 4] developed guidelines for selecting the Sim-Stim interface parameters that guarantee a specified level of probability of match between the stability of the PHIL system relative to the stability of the actual physical system. The type of power matching that the Sim-Stim interface must achieve depends on the specific power system (e.g., AC, DC, or AC/DC), its architecture, the operating conditions of interest (e.g., steady state or dynamic operation, normal/emergency) and

type of phenomena of interest (power quality, stability, etc.) As seen from this scenario, several key points are important for implementing this structure to connect remote power hardware together:

1) The data acquisition must be set up at a proper rate, so that accurate status information is collected. These data should be unambiguously marked with a time stamp.

2) The network must provide stable and time deterministic communication capability to deliver the status information quickly.

3) The time delay caused by data acquisition, A/D conversion and the communication would affect the accuracy of the simulation compared with the central hardware connection. So the sample rate and time delay should be coordinated and simulation stability should be analyzed as shown in papers [3, 4]

4) At the Sim/Stim interface, it constructs the voltage source and phantom load according to the received status information. Depending on the research focus of the experiment, the algorithm for the hardware controller could differ.

2.1.3 Network Capability for MURI

For a real-time simulation, the delay caused by data acquisition, A/D conversion and conversion communication may cause a stability problem [3,4]. Selection of hardware can control the data acquisition and A/D conversion delay. While the communication delay over the Internet is restricted by the load and network service provided, the network capability for real-time requirement needs to be considered.

For remote power system HIL test set up, real time access to remote instruments is the basic step for an integrated laboratory and simulation. Each lab is within the Local Area Network (LAN) of campus, and routers physically connect LANs over campuses.

First we should notice the LAN's limitation for providing real-time response. Since ethernet is the dominant LAN technology, researchers are interested in using it for their remote measurement and test environment. But ethernet uses the protocol CSMA/CD (Carrier Sense Multiple Access with Collision Detection) mechanism to control the Medium Access Control (MAC) layer access, thus data transmission is deferred for a different time when there is a collision. Thus, the ethernet is inherently non-deterministic. For effective real time applications researchers may need to resort to high bandwidth of Ethernet and fast Internet, coupled with the use of intelligent switches to deliver predictable, even deterministic input/output over the Ethernet [5].

Second, since most universities are members of Internet2, which provides router connection among campuses, they can use the advanced capability and facility provided by Internet2 can be used. At the physical layer, Internet2 has established an advanced network infrastructure with Backbones operating at 2.4 Gbps (OC48) to 10 Gbps (OC192) capacity. At the network layer, a new Quality of Service (QoS)----DiffServ, is applied in Internet2. This QoS ensures data with high priority could be delivered in time. Therefore, a high-speed network infrastructure exists to fulfill the real-time remote instrument access and control, and networked co-laboratories [6,7].

During development of our RTM applications, researchers use the QoS provided by the Internet2 through use of the QBone Scavenger Service. This service is a network

mechanism that allows applications to utilize otherwise unused network capabilities without adversely affected performance of the best-effort class of service.

2.2 Introduction of power system's computation problem

The research work at Mississippi State University focuses on the development of software tools to enable remotely distributed simulation. This research work looks at techniques to model and simulate a power system remotely at software level, as described in step one. Thus this section introduces the computation problem of power system.

The interconnected generation and transmission system is inherently large, and any problem formulation tends to have thousands of equations. The most common analysis, power flow, requires the solution of a large set of nonlinear algebraic equations, approximately two for each node. The usual algorithm of iterative matrix solutions exploits the extreme sparsity of the underlying network connectivity to gain speed and conserve storage. The power flow describes the steady state condition of the power network and thus, the formulation is a subset of several other important problems like the optimal power flow or transient stability [8].

In power system analysis, the transient stability program is used extensively for off-line studies but has been too slow for on-line use. A significant increase in speed will allow on-line transient stability analysis. The transient stability problem requires the solution of differential equations that represent the dynamics of the rotating machines together with the algebraic equations that represent the connecting network. This set of Differential Algebraic Equations (DAE) has various nonlinearities, and some sort of numerical method is usually necessary to obtain a step-by-step time solution. Each

machine may be represented by two to twenty differential equations, and so a 2000 bus power network with 300 machines may require 3000 differential equations and 4000 algebraic equations [8].

The size of these above problems and the consequent solution times encourages the search for distributed processing approaches. The concept of decomposing a large problem to address the time and storage problems has been applied to many of these power system problems. In fact, a rich literature exists of decomposition/aggregation methods.

2.2.1 Mathematical equation Formulation of power system

Basically, the solution of most power system problems requires the solution of the linear algebraic problem in the form,

$$Ax=b \tag{2.1}$$

where A is characterized as large with random sparsity, typically incidence symmetric and is often numerically symmetric. Also, x and b may or may not be sparse. A large number of direct and indirect algorithms can solve this problem. The most effective method on a serial processor for power system applications to date is the use of triangular factorization along with forward/backward substitution. For every square matrix A, there exists a LDU decomposition as shown in Equation (2.2).

$$LDU = A \tag{2.2}$$

Here, L represents a lower-triangular matrix with diagonal s of 1; D represents a diagonal matrix; and U represents a upper-triangular matrix with diagonal s of 1. The process of forward/backward substitution is shown in equation (2.3) and (2.4).

$$Ly = b \tag{2.3}$$

$$DUx = y \tag{2.4}$$

The two distinct phases to this problem are the factorization phase, (2.2), and the substitution phases, (2.3) and (2.4). The algorithms in which such solutions are required will dictate whether both phases can be processed simultaneously. For example, in a full Newton power flow, the Jacobian and mismatches are recreated on each iteration, so (2.2) must be solved repeatedly. The fast decoupled power flow requires that (2.2) be solved once and (2.3) and (2.4) be repeatedly solved in each iteration [9].

For the transient stability problem, the power network is defined by a set of nonlinear DAEs as follows:

$$\dot{y} = f(y, x) \tag{2.5}$$

$$0 = g(y, x) \tag{2.6}$$

where (2.5) describes the machine dynamics and (2.6) describes the network static behavior.

Sequential solution algorithms, developed over several decades, use two basic approaches. The first is the partitioned approach, where (2.5) is solved by an integration method (e.g. fourth order Runge-Kutta), and at every time step, (2.6) is solved separately. In the simultaneous approach, (2.5) is discretized (e.g. by the trapezoidal method) and then solved together with (2.6) at each time step using method like Newton. The latter approach tends to be faster because the sparse Jacobian can be held constant, and this variation is known as the very dishonest Newton (VDHN) method [8]. Also, the transient stability problem can be stiff, which means that an explicit method like the Runge-Kutta

may require very small time steps. Hence, they need longer computation time. To avoid numerical instability, an implicit method like the trapezoidal integration, which is inherently stable, is necessary [9].

2.3 Literature review of distributed simulation

Distributed simulation involves the use of networks of computers, which may not be located geographically close to one another. These computers communicate with one another through asynchronous channels whose speeds may vary over a wide range. It is the multiplicity of processes and the communication by message-passing that make the system distributed. There is no assumption about the relative speed of processes or the message transfer delays. This absence of timing assumptions makes the distributed system asynchronous. The growth of distributed simulation over centralized computing systems increases the computing power and improves reliability and computation speed. However, distributed simulation is more difficult to analyze and control than a centralized system, and geographical separation of the computers makes global memory unwieldy and asynchronous execution imperative.

The area of transient stability has attracted particular attention of distributed simulation because power engineers have a perceived need for on-line analysis to determine the security of the power system. The power flow problem has also been a target because of its pervasive use in the industry and also because of its ability to fit well with the matrix solution research being carried out outside the power area. The industry is demanding on-line applications in the dispatch center where fast computations give a power system operator more automatic analysis to help in decision-making. Distributed

simulation has the potential to boost the computation speed and make the analysis more interactive. That is, faster programs increase engineering efficiency.

Above all, the rapidly changing computation technology is providing the power system engineer with new ways for increasing cost and speed efficiency. The use of parallel and distributed computers to speed up calculations is attractive for the power industry.

2.3.1 Parallel algorithm

The use of a parallel algorithm can take advantage of these decomposition/aggregation techniques, but usually a certain amount of adaptation is necessary. Much of the research in applying parallel processing to power systems has its roots in this literature. In the past two decades a significant amount of research has been conducted in parallel processing of the power system problem [8]. Most of this work has been in the development of algorithms, while actual testing on multiprocessor architectures is more at the beginning stages. Most of the results, although encouraging, do not yet indicate clear cut paths for the development of production grade engineering tools. Thus far, the main thrust of research for parallel algorithms has been trying to address specific power system problems. The main goal in developing an algorithm is to maximize its parallelism and minimize the data dependencies [8].

To speed up solution of algebraic equations (1), researchers has done much work on algorithms for parallel triangular factorization [10-16] and/or forward and backward substitution [10-15,17,18]. Many of these algorithms have attempted to take the serial factorization/substitution problem and exploit available parallelism through

reordering/partitioning of the A matrix. This attempt has been effective in reducing the number of precedence relationships, which is governed by the maximum factor path length. The length of the longest factor path in the elimination tree seems to represent a fundamental limit in minimizing the number of precedence relationships during factorization [19]. Fundamentally, new algorithms attempting to minimize the precedence relationships with forward and backward substitution problems include the multiple factorization scheme [20] and the use of sparse inverse factors [18, 21, 22]. Other researchers have revisited indirect methods in an attempt to minimize the number of precedence relationships in solving (2.2) [23]. These algorithms put great effort on dealing with the precedence relationships inherent in the substitution phase. While algorithm development has yielded good theoretical results, little software has been developed for parallel machines to date. References [15, 24] report parallel factorization and substitution results on the Intel Processor iPSC, which are unimpressive. Full factorization can be accomplished with maximum speed gains on the order of two and with parallel gains of about 10 when factorization is halted before the densest portion of the matrix is encountered (partial factorization). Reference [25] has produced experimental results for solving (2.2) with a vector processor. These results show promise of the widely available vector machines usage in the algebra equations. These experimental results are helping to identify the real issues in parallel processing.

For the transient problem shown in equation (2.4) and (2.5), the parallel algorithm decomposes the system variables into groups and gets the solution for each group in parallel. In addition, since several time steps can be solved simultaneously, it is possible

to parallelize in time. The most obvious parallelization in space is the decomposition of equation (2.4) into sets of equations for each separate machine, and a master machine takes care of the interconnection being provided by (2.5). The first suggestion for distribution in time was made in [26], forming the Newton equations at each time step and solving simultaneously. However, implementation of these algorithms on an actual parallel computer has been slow because of hardware limitations.

2.3.2 Decoupled Circuit Method based on Graph

Contrary to parallel computation, another trend in large-scale circuit simulation has arisen using a decoupled circuit method based on graphs [27,28]. This method makes decoupling graphically, uses a simplified equivalent circuit to represent the external circuit, and then applies relaxation techniques iteratively to get the whole system simulation. This method has been implemented in large-scale circuit simulation since the early seventies. Until now, it has had the unique role in circuit simulation for saving simulation time and memory usage and hiding incompatible model descriptions. In this case, the algebraic and ODE system that describes a large circuit has to be decomposed into many loosely coupled subsystems. The solution to the subsystems is repeatedly recalculated by relaxing the behavior of the surrounding subsystems until the solution converges. However, poor convergence properties often limit the applicability of relaxation algorithms in large-scale circuit simulation. The problem arises when during the partitioning, the circuit is cut across the signal flow paths, and the resulting decomposed system cannot accurately reproduce the feedback existing in original circuit.

A number of partitioning methods have been proposed based on topology in papers [27-29]. They have two sets of constraints: if any tightly coupled nodes are not lumped together, then relaxation algorithm will converge very slowly, but if too many nodes are lumped together, the advantages of using relaxation will be lost. One improved approach is overlapping partitioning, where additional nodes and common elements are included into both the sub-circuits being separated. But this method increases the sizes of such parts and complicates the entire relaxation algorithm. In addition, the convergence is greatly affected by the decoupling point and the introduced element.

The basic coupling patterns are V (voltage)-type coupling and I (current)-type coupling. All relaxation-based circuit simulators use V-type coupling. To explain this method, the circuit in Figure 2.3 is taken as the example, which consists of two subsystems A and B. The subsystems have input conductance y_a and y_b respectively and connect through a conductance y_{ab}. Figure 2.3 represents the original system. Figure 2.4 represents the V-type coupling when the original circuit is decomposed through the conductance yab. x_1 and x_2 are the node voltages.

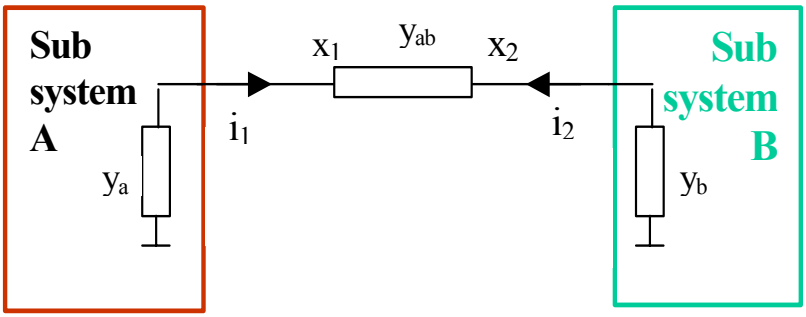


Figure 2.3 Original System before decoupling

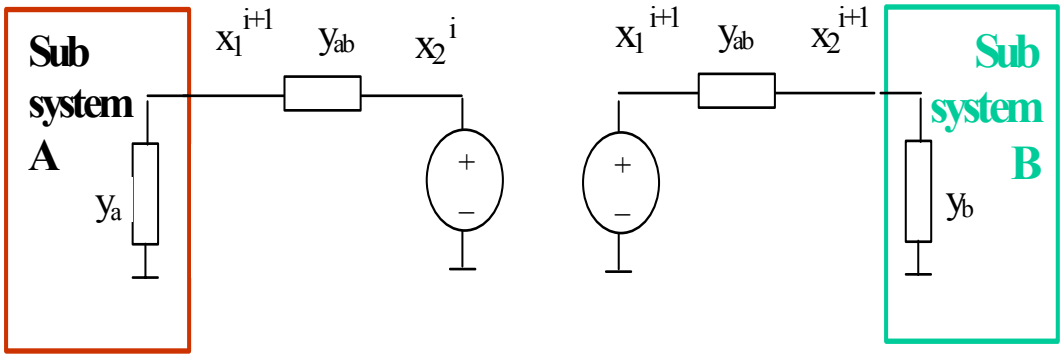


Figure 2.4 Decoupled System with V-Type Coupling [26]

Figure 2.4 shows two voltage sources controlled by x_2^i and x_1^{i+1}, and the conductive element y_{ab} presents in both parts. Noting that this coupling pattern produces a positive local feedback between x_1 and x_2 essential. Thus, a positive increment Δx_1^{i+1} results in the positive increment Δx_2^{i+1} in the part B, while the latter is conveyed at the next iteration to the first part and produces an additional positive increment Δx_1^{i+2}.

Figure 2.5 shows the second type of coupling, I-type coupling.

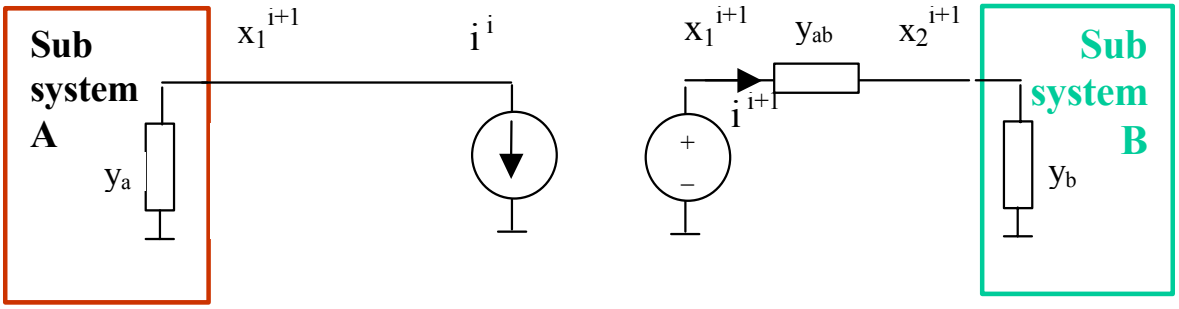


Figure 2.5 Decoupled System with I-Type Coupling [26]

This type of coupling is not very popular for the relaxation-based solver since it does not ensure convergence. Primarily, it couples parts connected by passive transistors. Both V-type and I-type coupling methods have problems in convergence. In papers

[27,28], Dr. Dmitriev-Zdorov presents a new approach to address the convergence problem caused by local couplings in the decomposed system. Figure 2.6 shows the coupling pattern. Both subsystems have controlled voltage source and controlled current source. An additional admittance y* is the stabilizing element in between the voltage and current source.

Such a coupling pattern changes the feedback factor to an optimal small number, based on the y* selection. Thus, the generalized VI (voltage-current) coupling pattern combines the properties of two basic couplings and may control the convergence by y*. This method has demonstrated its effectiveness in a mixed system simulation of MOS RAM cell. Similar circuit decoupling has been implemented for a large power electronic system [29]. It uses a long transmission line as the decouple point and represents the transmission line with Norton equivalent circuits in each sub-circuit.

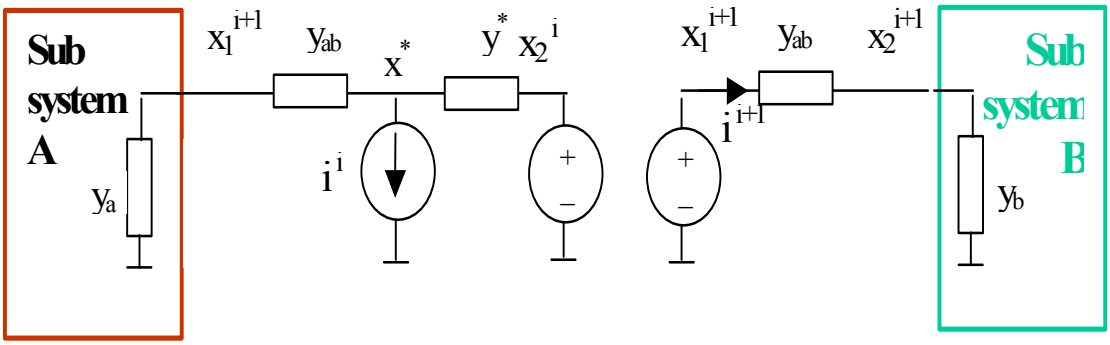


Figure 2.6 Decoupled System with VI-Type Coupling [26]

This algorithm employs a companion link model of transmission line to break a large circuit into many small sub-circuits for easy circuit formulation and fast simulation. This method is applicable to a system consisting of fast switching converters, decoupling

happening at the DC link part. A successful decoupled simulation technique, based on the Transmission Line Matrix (TLM), has confirmed that a multistage power electronic system can be achieved without sacrificing much accuracy. This algorithm is demonstrated in a converter-fed DC transmission system with a transmission-line length equivalent to 1000km in paper [29].

2.4 Summary

This chapter introduces the MURI project's background for this research, outlines the mathematical formulation of power system modeling and simulation are outlined, and discusses the computation complexity. It also discusses the existing methods used in distributed simulation and their characteristics are discussed.

CHAPTER III

STATEMENT OF PROBLEM AND WORK PLAN

This chapter summarizes the limitations of the existing method and introduces briefly technology and simulation tools for this research. It also presents a work plan for developing a new distributed simulation algorithm and load models for transient study.

3.1 Limitations of existing methods

Chapter 2 summarizes the methods for distributed simulation, including a parallel algorithm based on matrix operations and a graph based DC decoupling schema. Those algorithms developed for parallel processing are based on matrix operations and use divide and conquer methods to solve the system. The a master compute, which collects the overall network parameters and state variables, needs to formulate the system matrix in some degree. Based on the characteristics of the matrix, the mathematic problem is divided into smaller size portions, and separate slave computers solve the small portion. The algorithm concerns include computation load balance and communication delay. Thus, the hardware architecture affects the designation of parallel processing.

The decoupled circuit method has application for DC circuit analysis and mostly used for VLSI circuit analysis. In a decoupled simulation of a large power electronic system, decoupled points are selected at the DC link part. This method splits the circuit

graphically and avoids a complex matrix partition. At the same time, in the sub-circuit analysis, the admittance matrix sparsity can lead to simplifying the computation.

3.2 Proposed work

Conceptually, a large power system network can be considered as comprised of a number of sub-networks, or clusters, connected via tie-lines to a group of bus bars known as cut-nodes. Thus, spliting the system graphically into sub-networks and solving them individually is attractive. Also, when dealing with the system where the detailed model of some areas is hidden from others, dividing the system and solving the system based on the boundary value response is practical. This method allows not only a more simplified problem formulation process due to the small sizes of a decoupled network, but more importantly, it opens the way to parallel simulation.

This research work focuses on developing an agent based distributed simulation algorithm and a controllable load model for steady state analysis. I will work to extend the decoupled simulation method from a DC link to AC systems and explore the decoupled method for power system simulation. I will develop models for distributed simulation and implement it in time-domain simulation software package—Virtual Test Bed (VTB). different kinds of networks with different power sources and power load configurations demonstrate the algorithm's capability to deal with three-phase coupling.

3.3 Approaches to Problem

This section presentes, the simulation environment and basic technology used for distributed simulation.

3.3.1 Simulation Environment

Given the power system transient problem formulation as shown in differential/algebra equations (2.4) and (2.5) and the initial values and external inputs, ideally calculus techniques will find a nice analytical function for the state variables and outputs. For large-scale circuits and power systems problems, however, finding an analytical solution, and this method becomes impractical is impractical. Discretizing the differential/integral equations and using numerical methods will get the approximate solution. This technique is common for most computer aided circuit simulation software package such as Pspice, PSIM (produced by PowerSIM) and Virtual Test Bed (VTB). VTB was developed by the University of South Carolina and sponsored by ONR. It aims to provide a multidisciplinary simulation environment, which covers areas in electrical, thermal, chemical and mechanical engineering. It has an interactive simulation environment and advanced visualization capabilities [30,31].

The electrical models in VTB are classified as natural models and are developed using the Resistive Companion (RC) technique. Nodal analysis obtains the network solution. I selected VTB as the research environment for the reasons below:

- It is open architecture software and allows for user modeling with C++ coding or model transfer from other simulation software package such as Matlab, Labview and ASCL.
- It provides models for concurrent client/server model for distributed single-phase simulation, which is a good beginning for three-phase coupling study.

- It supports the run time change of system parameters, which makes it suitable for our MURI test plan.
- It has an extension to real-time VTB supporting hardware in the loop testing, which matches the ONR MURI project's final goal.

However, the algorithm for distributed simulation is not limited to VTB. It is applicable to all time domain simulation software.

3.3.2 Modeling techniques

VTB's modeling guide [24] describs the resistive companion network analysis technique. One of the merits of the RCF modeling technique is that physical conservation laws are enforced on natural coupling between neighboring components. This property makes the RCF model suitable for a power system network. The RCF technique can describe each object independently which also allows for easy object-oriented computer programming for a power system network simulator. Once a component model has been developed and tested in the VTB framework, the model can be implemented as a dynamically linked library (DLL) object. The released version can be used in any simulation of VTB [31].

In time domain simulators, the RCF modeling methodology has wide use. RCF is one of the most used techniques to describe a component in many circuit simulation software packages. Consider a power system component shown in Figure 3.1, it has a number of terminals that can be interconnected to other components [30].

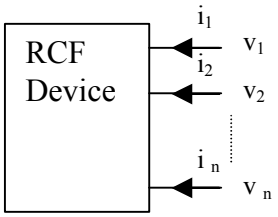


Figure 3.1 RCF Device Terminal's Definition

In a RCF expression, each terminal has an associated across variable and a through variable. For an electrical device, the across variables are the terminal voltage with respect to a common reference, and the through variables are the electrical currents flowing into the terminal respectively. Such a component can be described in the following general form using a set of algebraic and differential equations of [31]:

$$\begin{bmatrix} i \\ 0 \end{bmatrix} = \begin{bmatrix} f_1(\dot{v}, \dot{y}, ..., \int v, \int y, ..., v, y, u, t) \\ f_2(\dot{v}, \dot{y}, ..., \int v, \int y, ..., v, y, u, t) \end{bmatrix} \tag{3.1}$$

f_1, f_2 : Arbitrary vector functions; Function f_1 defines a set of external equations and function f_2 defines a set of internal equations.

i: a vector of through variables, it appears only in the external equations.

v: a vector of across (external) state variables.

y: a vector of internal state variables.

u: a vector of independent controls.

Similarly, the total number of equations in equation set (3.1) is the same as the total number of state variables, which includes external states and internal states.

In order to create an RCF model of this component, a discretized numerical integration method, which could be the trapezoidal rule or some other method, is applied to (3.1). For a linear device, it results in a set of equations like (3.2).

$$\begin{bmatrix} i(t) \\ 0 \end{bmatrix} = G \begin{bmatrix} v(t) \\ y(t) \end{bmatrix} - \begin{bmatrix} b_1(v(t-h), i(t-h), y(t-h)) \\ b_2(v(t-h), i(t-h), y(t-h)) \end{bmatrix} \tag{3.2}$$

Here, G is a Jacobian matrix and totally depends on the device parameters and the simulation time step in use. Functions b1 and b2 are history vectors that depend only on past history values of through, across, and internal states.

For a nonlinear device, the resulting algebraic equations must be solved by Newton's method iteratively at each time step. Hence, the RCF model is of the following form equation (3.3) at each time step:

$$\begin{bmatrix} i(t) \\ 0 \end{bmatrix} = G(v(t), v(t-h), y(t), y(t-h), t) \begin{bmatrix} v(t) \\ y(t) \end{bmatrix} - \begin{bmatrix} b_1(v(t), v(t-h), i(t-h), y(t), y(t-h), t) \\ b_2(v(t), v(t-h), i(t-h), y(t), y(t-h), t) \end{bmatrix} \tag{3.3}$$

Here, G is the Jacobian matrix of the discretized version of (3.1), and it will change with the solution of state variables at each current time step which causes the solution to be revised until it converges; functions b_1 and b_2 are vectors depending on the through variables, across variables, and internal states of the component at their current and previous time steps.

For generalization, the resistive companion equation is:

$$i(t)=G*v(t)-b(t-h), \tag{3.4}$$

where G is a constant matrix for linear devices and a dependent variable matrix for nonlinear device.

As an example, the illustration below demonstrates how to represent a capacitor into RCF model as shown in equation (3.4).

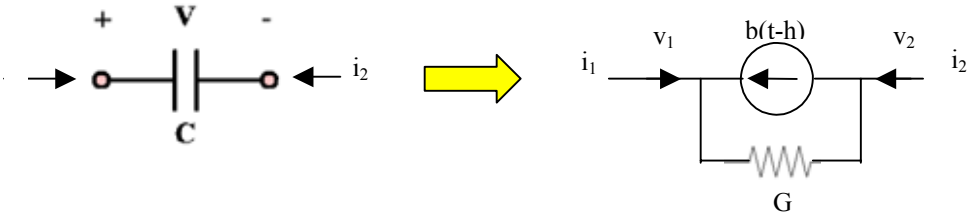


Figure 3.2 Representation of a Capacitor in RCF Model

A capacitor will follow the differential equation (3.5) and (3.6):

$$i_1(t) = C\frac{d(v_1 - v_2)}{dt} \tag{3.5}$$

$$i_2(t) = -i_1(t) \tag{3.6}$$

Applying the trapezoidal integration method to both sides of (3.5)

$$\frac{h}{2}(i_1(t)+ i_1(t-h)) = C((v_1(t)-v_1(t-h)) -(v_2(t)-v_2(t-h)))$$

$$i_1(t) = \frac{2C}{h}((v_1(t)- v_2(t)) - \frac{2C}{h}(v_1(t-h)-v_2(t-h))- i_1(t-h)$$

$$\begin{bmatrix} i_1(t) \\ i_2(t) \end{bmatrix} = \begin{bmatrix} \frac{2C}{h} & -\frac{2C}{h} \\ -\frac{2C}{h} & \frac{2C}{h} \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} - \begin{bmatrix} \frac{2C}{h}((v_1(t-h) - v_2(t-h)) + i_1(t-h)) \\ -\frac{2C}{h}((v_1(t-h) - v_2(t-h)) - i_1(t-h)) \end{bmatrix} \tag{3.7}$$

So comparing equation (12) to the generalized form, results in G= $\begin{bmatrix} \frac{2C}{h} & -\frac{2C}{h} \\ -\frac{2C}{h} & \frac{2C}{h} \end{bmatrix}$

$$\text{and } b(t\text{-}h) = \begin{bmatrix} \frac{2C}{h}((v_1(t-h) - v_2(t-h)) + i_1(t-h)) \\ -\frac{2C}{h}((v_1(t-h) - v_2(t-h)) - i_1(t-h)) \end{bmatrix}$$

When all of the components in the power system are modeled as RCF models, the circuit is changed into a DC circuit at the time instance. Nodal analysis can be applied to get the network solution. Energy conservation laws, such as Kirchhoff's Current Law (KCL), can be applied at each node of the system and result in the following set of equations:

$$\sum_k A^k i^k(t) = I_{inj} \tag{3.8}$$

where,

I_{inj} is a vector of nodal current injections,

A^k : a component incidence matrix

$$A_{ij}^k = \begin{cases} 1, & \text{if terminal j of component k connects to node i} \\ 0, & \text{otherwise} \end{cases}$$

i^k: the terminal through variables (currents) of component k

The component k terminal across variables $v^k(t)$ is related to the nodal vector of across variables v(t) by equation (3.9)

$$v^k(t) = (A^k)^T v(t) \tag{3.9}$$

Substituting $v^k(t)$ expression of (3.9) into the general device equation (3.10), results in

$$i^k(t) = G^k * v^k(t) - b^k(t-h) = G^k * (A^k)^T v(t) - b^k(t-h) \tag{3.10}$$

Substituting the $i^k(t)$ expression in (3.10) into (3.8), results in (3.11)

$$\sum_k A^k (G^k (A^k)^T v(t) - b^k (t-h)) = I_{inj} \tag{3.11}$$

v(t) can be solved from (3.10). If all Gs are linear, the network solution comes out in one iteration. Otherwise, at each time step, Newton's method solves these equations and iterative evaluation of the following expression gives the solution:

$$\begin{bmatrix} v(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} v^0(t) \\ y^0(t) \end{bmatrix} - \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} m_1^0 \\ m_1^0 \end{bmatrix} \tag{3.12}$$

where $v^0(t)$ and $y^0(t)$ are the values of the state variables at the previous iteration; m_1^0 and m_2^0 represents the mismatch of the system equations of the previous iteration; and the J matrix terms are block matrices of the system Jacobian matrix.

3.3.3 Agent technique and communication protocol

The term "agent" refers to a group of computer programs that are autonomous, with the ability to interact with other agents over network communications [32]. A multi-agent system is a kind of distributed computational system, where several agents take independent action and collaborate with other agents to achieve a certain goal. The agent technique is used in many areas of power system, including monitoring, diagnostic and reconfiguration. [32] introduces a monitoring and diagnostic platform based on agent technology. The agent-based technology enables the system to be flexible and easily reconfigurable. [33] proposes and demonstrates a multi-agent based algorithm for SPS reconfiguration with a simplified SPS.

Distributed simulation neess components are needed to collect/send information to remote sites and receive/reproduce the information at the local site. Agents here can

play a role to collect measurements at the natural coupling level as well as the signal coupling level. Also, the agents can be programmed to reproduce the response to local simulation with the environment through different equivalent techniques.

For agent communication, I selected the Microsoft Remote Procedure Call (RPC) facility to fulfill the communication needs between distributed simulations because the VTB model is written with C++ and uses MFC for user interface development. RPC invokes a function remotely through a standard interface. The functions interface called by RPC is defined by the Interface Definition Language (IDL), which is a standard language used to describe the interface to a routine or function. RPC can further migrate to Common Object Request Broker Architecture (CORBA), since objects in the CORBA are defined by an IDL. So, with simplified programming, RPC has the potential to extend to CORBA and allow more clients and a securer connection. Therefore, these properties provide the development tools to make an application easily adjustable within different network environments and not limit the distributed simulation algorithm within VTB only.

This section briefly introduces the concept of RPC and the working mechanism. RPC is designed to provide a common interface between applications and serves as a go–between for client/server communications. RPC can make client/server interaction easier and safer by factoring out common tasks, such as security, synchronization, and data flow handling, into a common library. RPC uses an interprocess communication (IPC) mechanism that enables data exchange and invocation of functionality residing in a different process. That process can be on the same computer, on the LAN, or across the

Internet. With RPC, essential program logic and related procedure code can exist on different computers. This property makes RPC suitable for the distributed simulations [34]. Also, RPC replaces dedicated protocols and communication methods with a robust and standardized interface. The functions contained within RPC are accessible by any program that must communicate using a client/server methodology. These properties provide the development tools to enable us to develop an application easily adjustable within different network environments.

Figure 3.3 shows the RPC process for the clients to call a procedure located in a remote server program. The client and server each have their memory resources allocated to data used by the procedure. The client side starts the RPC process. The client application calls a local stub procedure instead of having code implementing the procedure. Stubs are compiled and linked with the client application during development. Instead of containing code that implements the remote procedure, the client stub code retrieves the required parameters from the client address space and delivers them to the client runtime library.

The client runtime library then translates the parameters as needed into a standard Network Data Representation (NDR) format for transmission to the server [34].

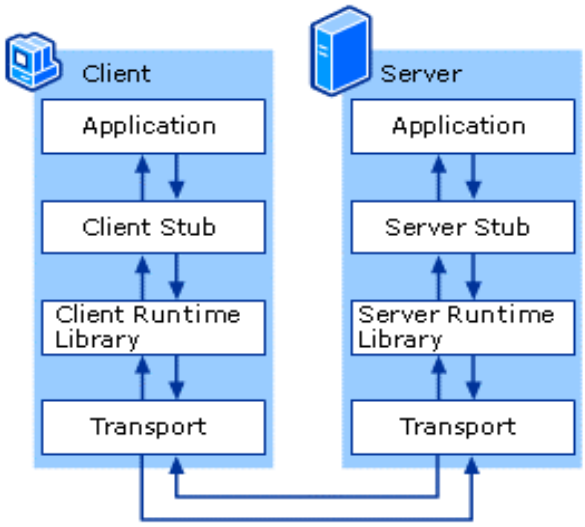


Figure 3.3 RPC Process for Calling a Procedure in a Remote Program [34]

The client runtime library then translates the parameters as needed into a standard Network Data Representation (NDR) format for transmission to the server [34].

The client stub then calls functions in the RPC client runtime library (rpcrt4.dll) to send the request and its parameters to the server. If the server is located on the same host as the client, the runtime library can use the Local RPC (LRPC) function and pass the RPC request to the Windows kernel for transport to the server. If the server is located on a remote host, the runtime library specifies an appropriate transport protocol engine and passes the RPC to the network stack for transport to the server. RPC can use other IPC mechanisms, such as named pipes and Winsock, to accomplish the transportation of the information. The other IPC mechanisms allow RPC more flexibility in the way in which it completes its communications tasks.

When the server receives the RPC, either locally or from a remote client, the server RPC runtime library functions accept the request and call the server stub

procedure. The server stub retrieves the parameters from the network buffer and, using one of the NDR marshalling engines, converts them from the network transmission format to the format required by the server. The server stub calls the actual procedure on the server. The remote procedure then runs, possibly generating output parameters and a return value. When the remote procedure is complete, a similar sequence of steps returns the data to the client. The remote procedure returns its data to the server stub which, using one of the NDR marshalling engines, converts output parameters to the format required for transmission back to the client and returns them to the RPC runtime library functions. The server RPC runtime library functions transmit the data to the client computer using either LRPC or the network.

The client completes the process by accepting the data over the network and returning it to the calling function. The client RPC runtime library receives the remote-procedure return values, converts the data from its NDR to the format used by the client computer, and returns them to the client stub. The server application contains calls to the server runtime library functions, which register the server's interface with the RPC runtime. The server application also contains the application-specific remote procedures that are called by the client applications. Using RPC, developers can transparently communicate between different types of processes; RPC automatically manages process differences behind the scenes.

3.4 Work Plan

This research begins with a derivation of a load model, which is a commonly used element for power system analysis. This task is in three steps:

- Develop a PQ load, which allows user to specify PQ parameters.
- Develop a controllable PQ load, which have signal ports to control the PQ.
- Validate the PQ load model with other simulation software in transient analysis and steady state analysis.

After the construction of load model, I propose a distributed algorithm to deal with natural coupling and signal coupling. This task can be divided into three steps:

- Propose distributed algorithm and perform stability analysis.
- Develop natural coupling model and perform accuracy analysis for different network configurations.
- Develop signal coupling model and perform accuracy analysis.

The next chapter provides details on the derivation of the load models.

3.5 Summary

In this chapter discusses the limitations of existing methods for distributed simulation. It justifies the motivation for developing a new distributed algorithm and introduces the supporting technology and simulation tools for this research. It also presents a work plan for this research.

CHAPTER IV

SINGLE PHASE LOAD MODEL

This chapter mainly describes the development of the load model, beginning with a conventional polynomial model, then proceeding to a RLC based PQ load model. Both load models' mathematical models are derived and implemented in VTB. It presents and analyzes the load models performance in steady state and transient study.

4.1 Introduction

Power system performance during transient operating conditions is of great interest for protection design and stability analysis. Traditionally, the short circuit is solved using a phasor based simulation program like ASPEN or PowerWorld. Then, the phasor solutions of prefault, fault, and post fault are transferred into time domain waveforms and combined together. The combined waveforms are used as the test signal for protection devices. However, this degree of simplification ignores the transient in the network when a fault occurs. A more appropriate waveform can only be generated through time domain simulation. To fulfill this function, the phasor models need to be converted into a time domain expression.

The load model is one of the most used phasor models in power system analysis. Generally, power system loads are categorized as constant impedance load, constant current load, and constant power load. In a practical power system, a composite of these

three types may be needed to represent the load and may be modeled as a polynomial load [35]. In order to properly simulate transient conditions in a power system, selecting appropriate load models that accurately describe the load is important. The load models should be sufficiently representative from the point of view of facilitating reliable transient analysis results. In order to enable different kinds of three-phase load modeling, such as a balanced or unbalanced load, and a grounded or ungrounded load, a single-phase load can be used as a basic element. Then, a balanced or unbalanced load and grounded or ungrounded load can be constructed through different connections of single-phase load models. Note that power loads are generally frequency-dependent and should also be modeled appropriately.

This chapter demostrates two approaches to develop load models. First, it preseetns a polynomial load model for single-phase implemented as a nonlinear load whose impedance changes according to the RMS measurement of node voltage. Second, it presentes a single-phase PQ load model based on constant impedance, which represents a linear load as a parallel or series combination of R, L and C elements. The load is characterized by the amount of real power (P) and reactive power (Q), which is a linear function of the square of the load voltage. Such a load has constant impedance characteristics assuming a constant frequency.

The new load models are implemented in a time-domain power system modeling and simulation environment, the Virtual Test Bed, which has been developed at the University of South Carolina [36]. Details about VTB's capabilities and environment can be found in [37]. I derived the mathematical resistive companion models for both load

models. The models are then validated through the comparison of VTB simulation, PowerWorld and Matlab/SimPowerSystem simulation. The results prove to be satisfactory.

4.2 Mathmatical formulation

4.2.1 Generic polynomial power load model

One of the most frequently used load models, the polynomial load model, is expressed in the following quadratic expressions [35]:

$$P = P_0 * [a(\frac{V}{V_0})^2 + b(\frac{V}{V_0}) + c] \tag{4.1}$$

$$Q = Q_0 * [a(\frac{V}{V_0})^2 + b(\frac{V}{V_0}) + c] \tag{4.2}$$

where a, b and c are the ratio of different types of load, P_0 and Q_0 are the real and reactive power consumed by the load at the reference voltage V_0. V is the RMS value of voltage at the load terminal. This model can be a combination of constant impedance load (denoted as Z), constant current load (denoted as I), and constant power load (denoted as P). For example, if a=0.5, b=0.3, c=0.2, then the load will have 50% of Z, 30% of I, and 20% of P. In this load model, only V is a time variant parameter, and the other parameters are static. In time domain simulation, V is an accumulative value and is calculated through the sliding window method [38]. (4.3) shows the formula below:

$$\vec{V} = \frac{\sqrt{2}}{N} \sum_{j=K}^{K+N-1} v(j * \Delta t) * e^{-j\frac{2\pi i}{N}} \tag{4.3}$$

Here, N is the total sample number for one cycle, and K is the start sample point of last cycle. Δt is the time step for the simulation. For steady state analysis, the $\vec{V}$ is calculated until the last sample point can be taken as the V for the current point. For a small time step simulation, this approximation will not introduce much error. Thus, V can be turned into a known value based on history during simulation time.

Thus, I selected the following properties, as shown in Table 4.1, as parameters.

Table 4.1

Parameter List of Polynomial Load

Parameter	Description
V_0	The nominal voltage of the load, in volts RMS
f_0	The nominal frequency, in hertz.
P_0	The active power of the load, in watts. Specify a positive value, or 0.
Q_0	The inductive reactive power Q_L, in vars. Specify a positive value, or 0.
A	The percentage of constant impedance load
B	The percentage of constant current load
C	The percentage of constant power load

Then, the P and Q at the time point are known, based on (4.1), (4.2) and (4.3). The load model is translated into impedance that consumes the PQ as indicated and applicable for that time point. The R and L are derived using the equations below:

$$R = \frac{V^2 P}{P^2 + Q^2} \tag{4.4}$$

$$L = \frac{V^2 Q}{2\pi f (P^2 + Q^2)} \tag{4.5}$$

The derived VTB resistive companion model is in the following form:

$$i(t) = G(t) * v(t) - b(t-h) \tag{4.6}$$

Using to KVL, results in the equation (4.7) below:

$$R*i+L\frac{di}{dt}=V_0-V_1 \tag{4.7}$$

After applying trapezoidal integration to (4.7), i(t) is expressed in RCF:

$$i(t)=\frac{\frac{h}{2}}{\frac{h}{2}R+L}v_0(t)-\frac{\frac{h}{2}}{\frac{h}{2}R+L}v_1(t)-b_0(t-h)$$

$$b_0(t-h)=\frac{-\frac{h}{2}}{\frac{h}{2}R+L}v_0(t-h)+\frac{\frac{h}{2}}{\frac{h}{2}R+L}v_1(t-h)+\frac{\frac{h}{2}R-L}{\frac{h}{2}R+L}i(t-h)$$

Thus,

$$G(t)=\begin{bmatrix} \frac{\frac{h}{2}}{\frac{h}{2}R+L} & -\frac{\frac{h}{2}}{\frac{h}{2}R+L} \\ -\frac{\frac{h}{2}}{\frac{h}{2}R+L} & \frac{\frac{h}{2}}{\frac{h}{2}R+L} \end{bmatrix}$$

4.2.2 RL- based load

Besides the polynomial model, which is usually an inductor motor load, in power system analysis, another type of load exists—constant RLC based load. RLC-based load is constructed by R, L and C, either in series or in parallel. This section derives a RCF model for a RLC-based PQ load. To enable different kinds of three-phase load modeling,

such as balanced/unbalanced load and grounded/ungrounded load, a single-phase PQ load model can be used as a basic element. This RLC based PQ load model accepts the following parameters, shown in Table 4.2.

Table 4.2

Parameter List of RLC Based PQ Load

Parameter	Description
V_{rated}	The nominal voltage of the load, in volts RMS
f_n	The nominal frequency, in hertz.
P	The active power of the load, in watts. Specify a positive value, or 0.
Q_L	The inductive reactive power Q_L, in vars. Specify a positive value, or 0.
Q_C	The capacitive reactive power Q_C, in vars. Specify a positive value, or 0.
bParallel	The flag for RLC configuration. When it is true, the equivalent RLC are assumed to be in parallel. When it's false, the equivalent RLC are assumed to be in series.

During the model derivation, the load model is treated as a two terminal device. v_0 and v_1 denote the terminal voltage associated with the corresponding terminals; i_0 and i_1 represent the current flow through the associated terminals. The general form of the RCF model is in (4.8):

$$i(t) = G * v(t) - b(t-h) \tag{4.8}$$

For a resistor, the RCF model is

$G_R = \frac{1}{R}$

$b_R(t-h) = 0$

For an inductor, the RCF model is

$$G_L=\frac{h}{2L}$$

$$b_L(t-h)= -\frac{h}{2L}(v_0(t-h)-v_1(t-h))-i_L(t-h)$$

For a capacitor, the RCF model is

$$G_C=\frac{2C}{h}$$

$$b_C(t-h)= \frac{2C}{h}(v_0(t-h)-v_1(t-h))+i_C(t-h)$$

Here, assume $\omega = 2*\pi *f_n$ (i.e., constant frequency) for all the derivations below. A PQ load has two possible combinations: parallel and serial.

1) The formula below calculates RLC parameter for the parallel case:

$R= V_{Rated} * V_{Rated} /P;$

$L= V_{Rated} * V_{Rated} /Q_L/\omega;$

$C=Q_C/(V_{Rated}*V_{Rated})/\omega;$

Thus, consider that the G and history current b(t-h) is obtained by adding up the admittance and history currents of each element. Also, notice that P and Q_L appear in the denominator in R's and L's expressions. Hence, special cases are properly develped for conditions when P and/or Q_L equal 0. All mathematical models for the four possible conditions in parallel are below:

a) When $P \neq 0$, $Q_L \neq 0$, $Q_C \neq 0$,

$$G = \frac{1}{R}+\frac{2C}{h}+\frac{h}{2L}$$

$$b_0(t\text{-}h) = -\frac{h}{2L}(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))\text{-}i_L(t\text{-}h) + \frac{2C}{h}(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))+i_C(t\text{-}h)$$

$$i_L(t\text{-}h) = -\frac{h}{2L}(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))\text{-}i_L(t\text{-}2h)$$

$$i_C(t\text{-}h) = \frac{2C}{h}(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))+i_C(t\text{-}2h)$$

b) When $P = 0$, $Q_L \neq 0$, $Q_C \neq 0$,

$$L = V_{Rated} * V_{Rated} / Q_L / \omega ;$$

$$C = Q_C / (V_{Rated} * V_{Rated}) / \omega ;$$

$$G = \frac{2C}{h} + \frac{h}{2L}$$

$$b_0(t\text{-}h) = -\frac{h}{2L}(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))\text{-}i_L(t\text{-}h) + \frac{2C}{h}(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))+i_C(t\text{-}h)$$

$$i_L(t\text{-}h) = -\frac{h}{2L}(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))\text{-}i_L(t\text{-}2h)$$

$$i_C(t\text{-}h) = \frac{2C}{h}(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))+i_C(t\text{-}2h)$$

c) When $P \neq 0$, $Q_L = 0$, $Q_C \neq 0$,

$$C = Qc / (V_{Rated} * V_{Rated}) / \omega ;$$

$$G = \frac{2C}{h}$$

$$b_0(t\text{-}h) = \frac{2C}{h}(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))+i_C(t\text{-}h)$$

$$i_C(t\text{-}h) = \frac{2C}{h}(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))+i_C(t\text{-}2h)$$

d) When $Q_L = 0$, $P \neq 0$, $Q_C \neq 0$,

$$G = \frac{1}{R} + \frac{2C}{h}$$

$$b_0(t\text{-}h) = \frac{2C}{h}\,(v_0(t\text{-}h)\text{-}v_1(t\text{-}h))+i_C(t\text{-}h)$$

$$i_C(t\text{-}h) = \frac{2C}{h}\,(v_0(t\text{-}2h)\text{-}v_1(t\text{-}2h))+i_C(t\text{-}2h)$$

2) For the series case, the RLC parameters scalculated using the formula below:

$$I_{cal} = \frac{\sqrt{P^2 + (Q_L - Q_C)^2}}{V_{Rated}};$$

$$R = \frac{P}{I_{cal}^{\ 2}};$$

$$L = \frac{Q_L}{I_{cal}^{\ 2}\omega};$$

$$C = \frac{I_{cal}^{\ 2}\omega}{Q_C}.$$

Similarly, noticing Q_C appears in the denominator in C's expressions, a special case is also accommodated for a condition when Q_C equals 0. All mathematical models for two possible conditions in series are listed below:

a) When $P \neq 0$, $Q_L \neq 0$, $Q_C \neq 0$,

Using to KVL results in the equations (4.9) and (4.10):

$$R*i+L\frac{di}{dt}+V_C=V_0-V_1 \qquad (4.9)$$

$$C\frac{dV_C}{dt} = i \qquad (4.10)$$

Applying trapezoidal integration allows to express *i*(t) in RCF:

$$i(t) = \frac{\frac{h}{2}}{\frac{h}{2}R + L + \frac{h^2}{4C}} v_0(t) - \frac{\frac{h}{2}}{\frac{h}{2}R + L + \frac{h^2}{4C}} v_1(t) - b_0(t-h)$$

$b_0(t-h) =$

$$\frac{-\frac{h}{2}}{\frac{h}{2}R+L+\frac{h^2}{4C}} v_0(t-h) + \frac{\frac{h}{2}}{\frac{h}{2}R+L+\frac{h^2}{4C}} v_1(t-h) + \frac{\frac{h}{2}R-L+\frac{h^2}{4C}}{\frac{h}{2}R+L+\frac{h^2}{4C}} i(t-h) + \frac{h}{\frac{h}{2}R+L+\frac{h^2}{4C}} v_c(t-h)$$

$$v_c(t-h) = \frac{h}{2C} i(t-h) + \frac{h}{2C} i(t-2h) + v_c(t-2h)$$

b) When $P \neq 0$, $Q_L \neq 0$, $Q_C = 0$

Equation (4.11) is derived according to KVL below:

$$R*i+L\frac{di}{dt} = V_0 - V_1 \qquad (4.11)$$

Applying trapezoidal integration, $i(t)$ in RCF is:

$$i(t) = \frac{\frac{h}{2}}{\frac{h}{2}R + L} v_0(t) - \frac{\frac{h}{2}}{\frac{h}{2}R + L} v_1(t) - b_0(t-h)$$

$$b_0(t-h) = \frac{-\frac{h}{2}}{\frac{h}{2}R + L} v_0(t-h) + \frac{\frac{h}{2}}{\frac{h}{2}R + L} v_1(t-h) + \frac{\frac{h}{2}R - L}{\frac{h}{2}R + L} i(t-h)$$

4.2.3 Controllable load model

As of now, two types of single-phase PQ load have been developed. However, to enable the MURI remote testing and measurement, the load models need to be extended

to be controllable and remotely accessible. To make a load controllable in the VTB modeling context, signal ports must be added to the load models. Thus, in the model development, the mathematical formulation will not change. Only the source of control data changes that is used for setting the conductance matrix G and history vector b in VTB context.

In addition, in VTB, the parameter setting is the static setting in the simulation. The parameters are read only once when initialization. By contrast, signal is dynamic and could be generated from a complex control function. Therefore, it needs to be read at every step, and the conductance matrix G needs to be changed whenever the signal changes.

To make load models controllable, control data must be determined. For a polynomial load, I selected a, b and c as external control. For RLC-based PQ load model, I selected P, Q_L and Q_c are as external control data. The external control achieves control by changing to signal port data in VTB model programming.

4.3 Test cases and performance analysis

4.3.1 Polynomial Load in Steady state analysis

Certain test cases verify the correctness of the developed polynomial load model in VTB. The steady state result is compared to the result from PowerWorld simulation and the power flow program.

A. Test case #1

The first test case is a two-bus system with load combination of constant PQ, constant current and constant impedance. Figure 4.1 shows the test circuit in VTB and Figure 4.2 shows the test circuit in Power World respectively.

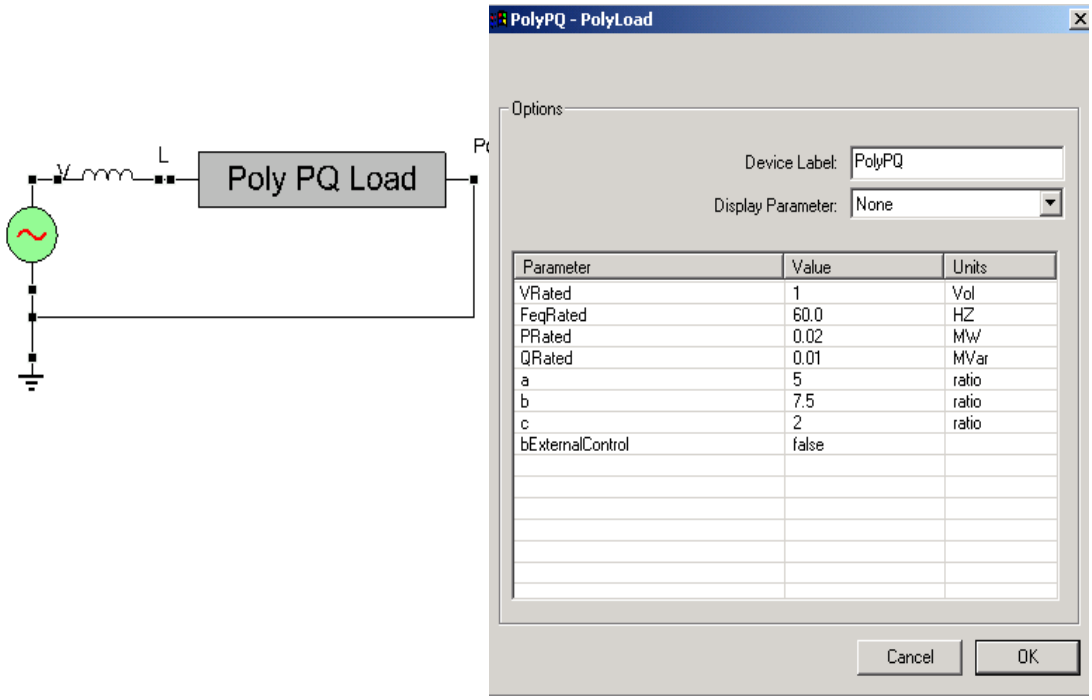


Figure 4.1 Two-bus Simulation in VTB

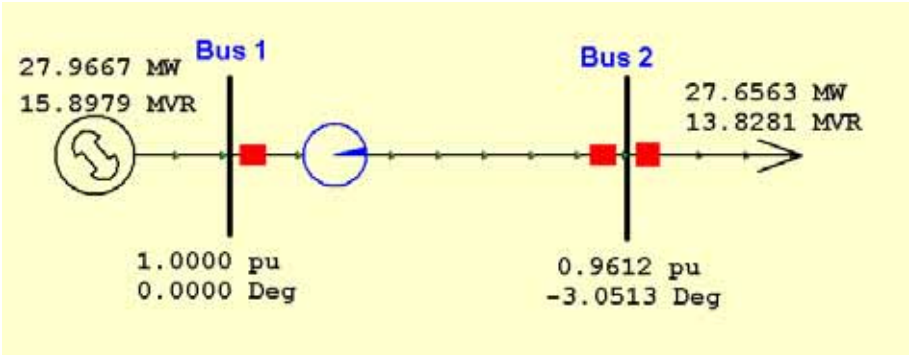


Figure 4.2. Two-bus Simulation in Powerworld

Table 4.3 shows the system specification data for this test case. Table 4.4 shows the steady state results.

Table 4.3

System Data For Test Case #1

Generator	$1\angle 0$ pu		
Transmission Line	0.03000+0.2i pu		
Load	P_0 MW)	Q_0 (MVar)	V_0 (pu)
	29	14.5	1
	A	b	c
	0.1379	0.5172	0.3448

Table 4.4

Result Comparison For Test Case #1

Result	PowerWorld	VTB
V_1 (pu)	1.0000	1.0000
δ_1 (deg)	0.0000	-0.0000
V_2 (pu)	0.96	0.96
δ_2 (deg)	-3.05	-3.05
P_G (MW)	27.97	28.77
Q_G (MVar)	15.89	14.39
P_L (MW)	27.66	27.65
Q_L(MVar)	13.83	13.83

As observed from Table 4.4, the voltage profile matches well with VTB simulation and PowerWorld simulation. The generators P and Q shows some mismatch,

which could be caused by the generator modeling. In PowerWorld, bus 1 is taken as the slack bus, while in VTB simulation it is an ideal voltage source with a series resistance. The Fourier transform and truncation contribute to the mismatch in the calculation. Also, the power flow program tolerance and Jacobian formulation selection will affect the accuracy of solution.

B. Test case #2

To demonstrate that the load model is complementary with steady state power flow, a more complicated test case was developed based on a new benchmark test system of a shipboard distribution network described in paper [39].

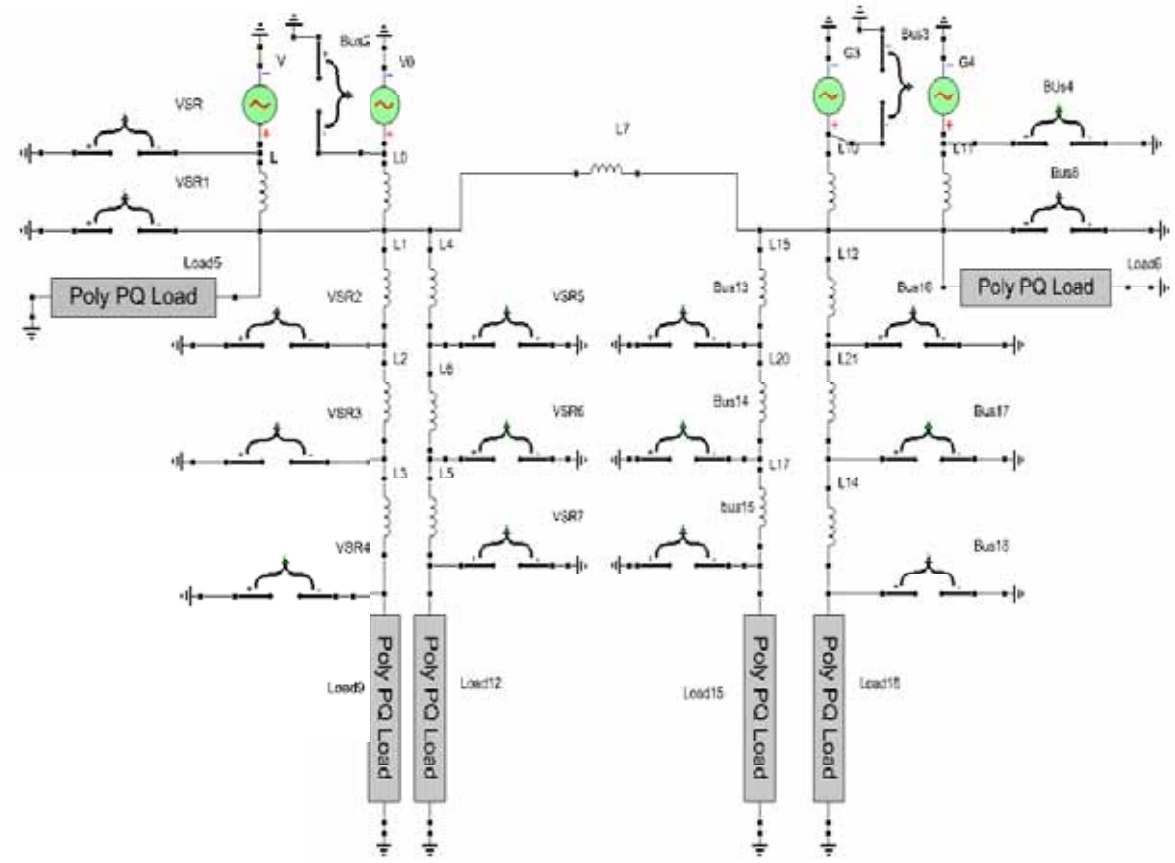


Figure 4.3 18-bus Shipboard Power System

Table 4.5

Power Flow Solution Comparison of VTB and Power Flow Program

Bus No.	Voltage magnitude (pu)		Voltage angle (degree)		Real Power Generation, P (MW)		Reactive Power Generation, Q (MVar)	
	VTB	PF	VTB	PF	VTB	PF	VTB	PF
1	1.02	1.020	0	0.000	5.722	5.808	1.292	1.218
2	1.02	1.020	0	0.000	6.329	6.150	1.424	1.567
3	1.02	1.020	0	0.000	6.112	6.040	1.391	1.447
4	1.02	1.020	0	0.000	5.896	6.060	1.335	1.202
5	1.0199	1.020	-0.0054	-0.006	-0.42	-0.42	-0.31	-0.31
6	1.0199	1.020	-0.0053	-0.005	-0.38	-0.38	-0.29	-0.29
7	1.0197	1.020	-0.0136	-0.014				
8	0.9866	0.987	-10.5502	-10.536				
9	0.9865	0.987	-10.5638	-10.549	-5.72	-5.72	-0.12	-0.12
10	1.0198	1.020	-0.0111	-0.011				
11	0.9873	0.987	-10.6174	-10.603				
12	0.9872	0.987	-10.6311	-10.616	-5.76	-5.76	-0.09	-0.09
13	1.0196	1.020	-0.0167	-0.017				
14	0.9872	0.987	-10.4745	-10.460				
15	0.9870	0.987	-10.4911	-10.477	-5.68	-5.68	-0.11	-0.11
16	1.0197	1.020	-0.015	-0.015				
17	0.9851	0.985	-10.7343	-10.719				
18	0.9849	0.985	-10.7502	-10.737	-5.81	-5.81	-0.14	-0.14

This work modeled the 18-bus shipboard power system in VTB. The simulation lasts one second to reach the steady state. Figure 4.3 shows the simulation. The data are

post-processed to get the RMS value of voltage profile and PQ at generator side and load side. Table 4.5 shows the power flow solution using VTB.

Compared to the results from the power flow solution in paper [39], the voltage magnitude is the same while the voltage angle deviation is within 0.014 degrees. Similar to test case 1, there is P and Q mismatch in the generators. The close voltage match indicates that the simulation conforms to the power flow analysis and can be used for transient analysis.

4.3.2 RLC-based PQ Load in transient analysis

The RLC-based PQ load model is coded in C++ programming language and compiled as a VTB Dynamic Link Library (DLL) component. The simulation in Figure 4.4 shows tests for different types of single-phase PQ loads.

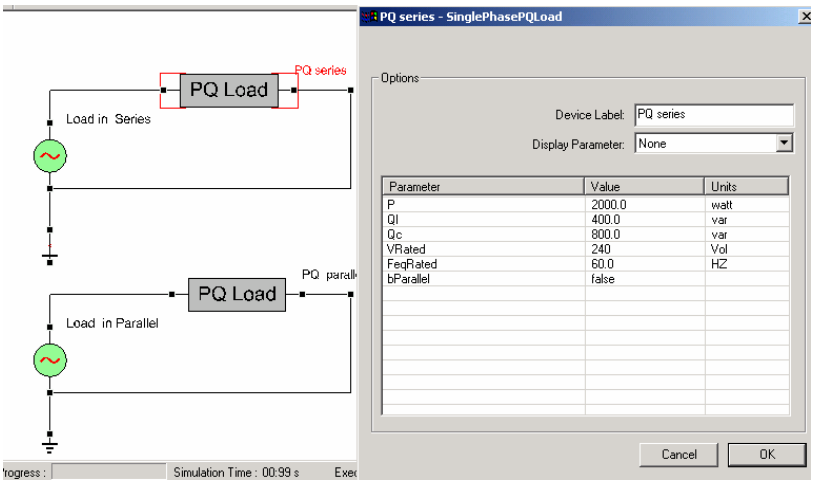


Figure 4.4 VTB Schematic For Single-Phase PQ Load Model Testing

In order to validate the newly developed VTB load model, Figure 4.5 shows the corresponding circuits, "SerialLoad.mdl" and "parallelLoad.mdl", in Matlab SimPowerSystem blockset.

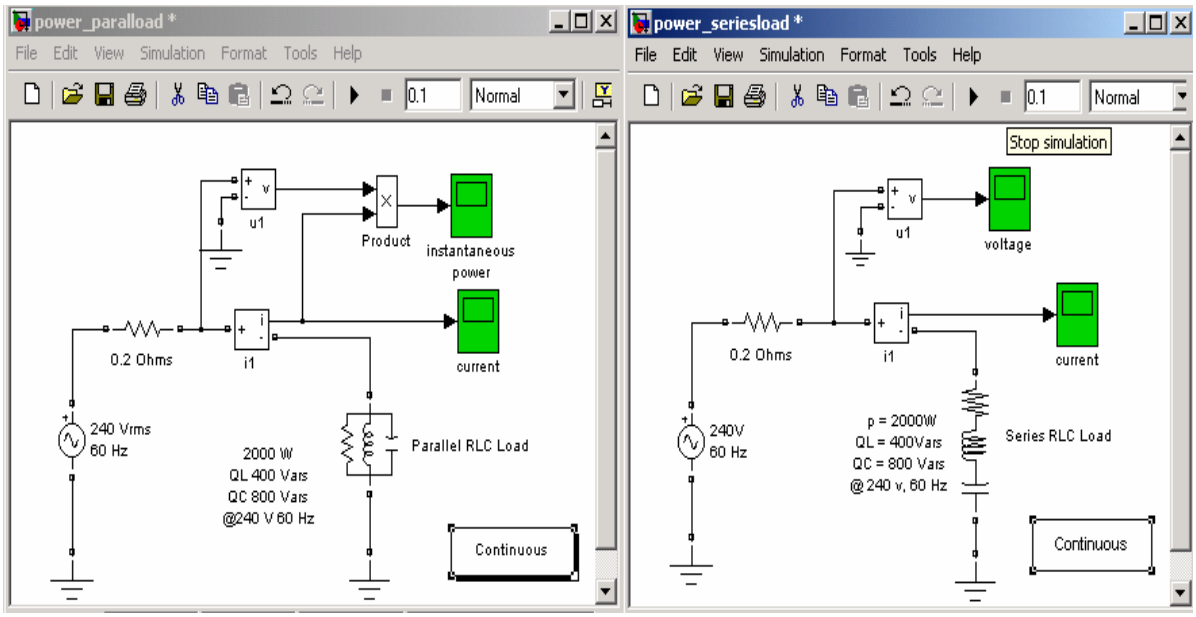


Figure 4.5 Matlab/SimPowerSystem for PQ Load Model Testing

Table 4.4 lists the test system specifications (the voltage source and load). The parameters are selected to cover the most complicated case in which all R, L and C elements are present.

Figures 4.6 and 4.7 display the comparison of the current waveforms obtained from VTB simulation and from Matlab/SimPowerSystem simulation. Specifically, Figure 4.6 and 4.7 gives the current flow through the series load and parallel load, respectively. The same time step of 10 μ s is used in both simulations.

Table 4.6 System Specifications for RLC Based Load Test

Parameter	Value
P	2000 W
Q_L	400 Var
Q_C	800Var
V_{Rated}	240V
f	60Hz
Rs	0.2 Ohms

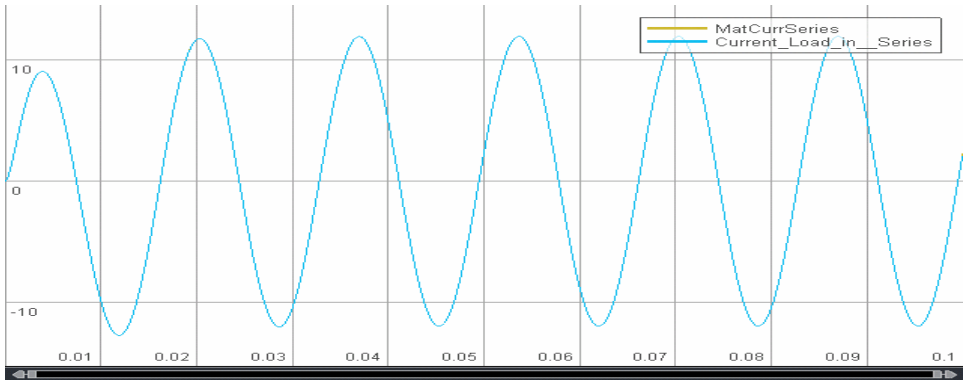


Figure 4.6 Current Waveforms for Load in Series

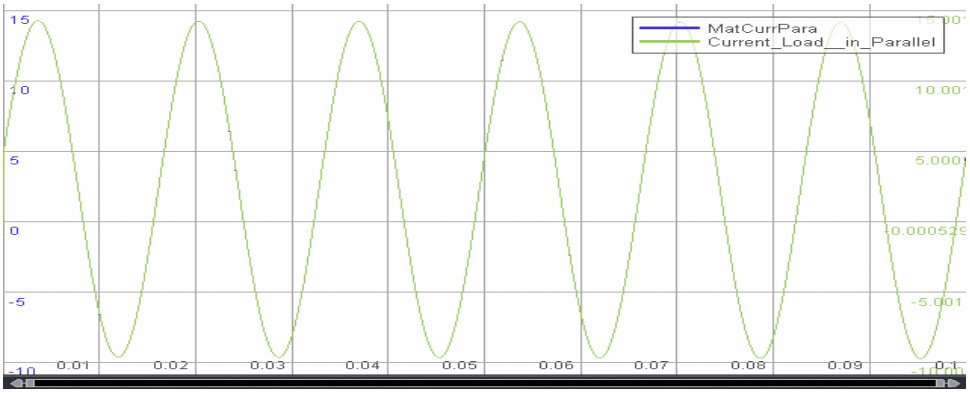


Figure 4.7 Current Waveforms for Load in Parallel

Figures 4.6 and 4.7 show, generally, the overall waveforms match the results of the Matlab simulation well.

The analytical solution for a series load can be expressed as

$i(t)=Ae^{-bt}\sin(\omega t)$

where A, b and c are decided by the R, L and C obtained from the calculation. It approximates a sinusoidal function.

When viewing the waveform mmre closely, the mismatch caused by discretization and the integration method used can be observed, shown in the enlarged series load waveform in Figure 4.8. The mismatch is because VTB's trapezoidal integration is an implicit one, while Simulink's ordinary differential equation 3 (Ode3) integration is an explicit one. The former integration method leads to a more stable solution and introduces less error but costs more in computation time. The later integration method costs less in computation time, while the solution is not absolutely stable and introduces more error.

These waveforms from VTB are obtained for the time step set at $10\,\mu$s. When it decreases, the dashed line will go closer to the sinusoidal curve and less mismatch is observed.

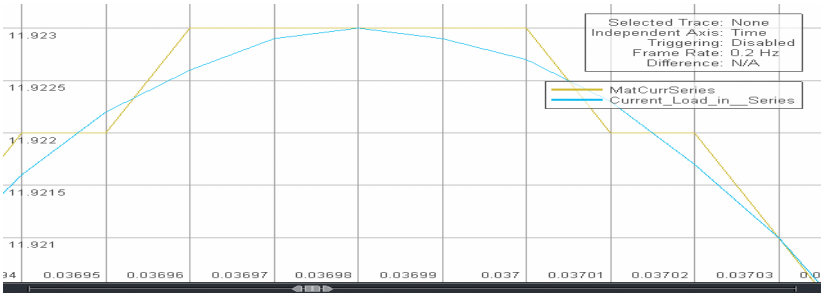


Figure 4.8 Enlarged Current Waveforms for Load in Series

Thus, these comparisions validate the new RLC-based PQ load; it can be used in time domain power system analysis.

4.3.3 Signal Controllable Load test

To test the signal controllable load, I add signal ports are added to the load model and implement them in VTB as a Dynamic Link Library (DLL) component. For RLC based PQ load model, P, Q_L and Q_c are now come from to signal port. Figure 4.9 shows the test circuit. RLC-based PQ loads are tested with parallel and series connection at the same signal input. Figures 4.10 and 4.11 showss the comparision of the signal controllable test results and the simulation without signal control. Table 4.3 shows the system specification. An additional inductance is introduced between power source and load to make it more realistic. The inductance is with L as 5.9130 mH and R as 0.34 Ohms, as used in the MURI project.

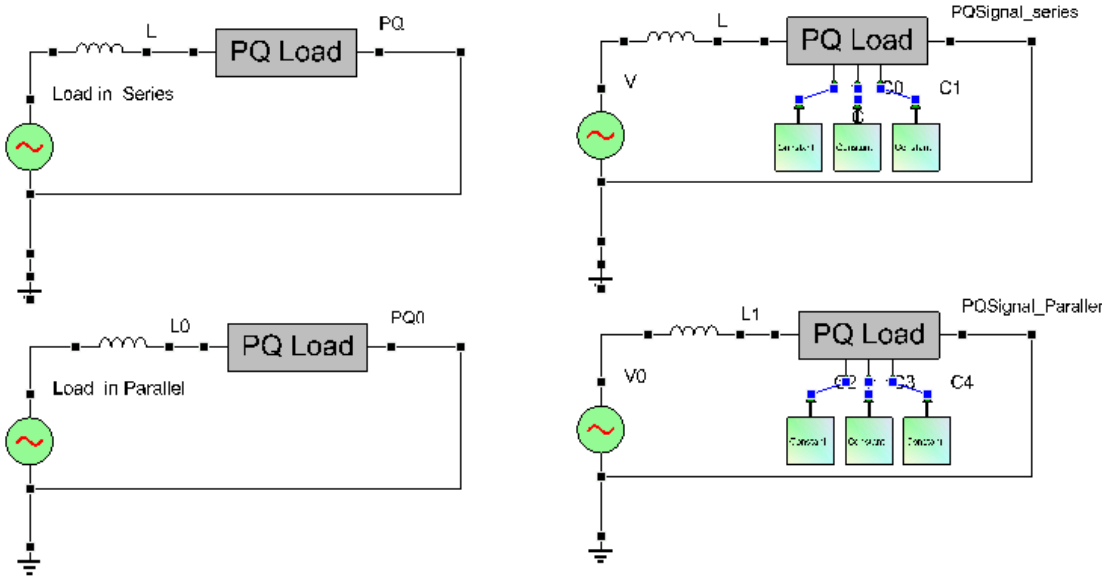


Figure 4.9 Matlab/Simpowersystem for PQ Load Model Testing

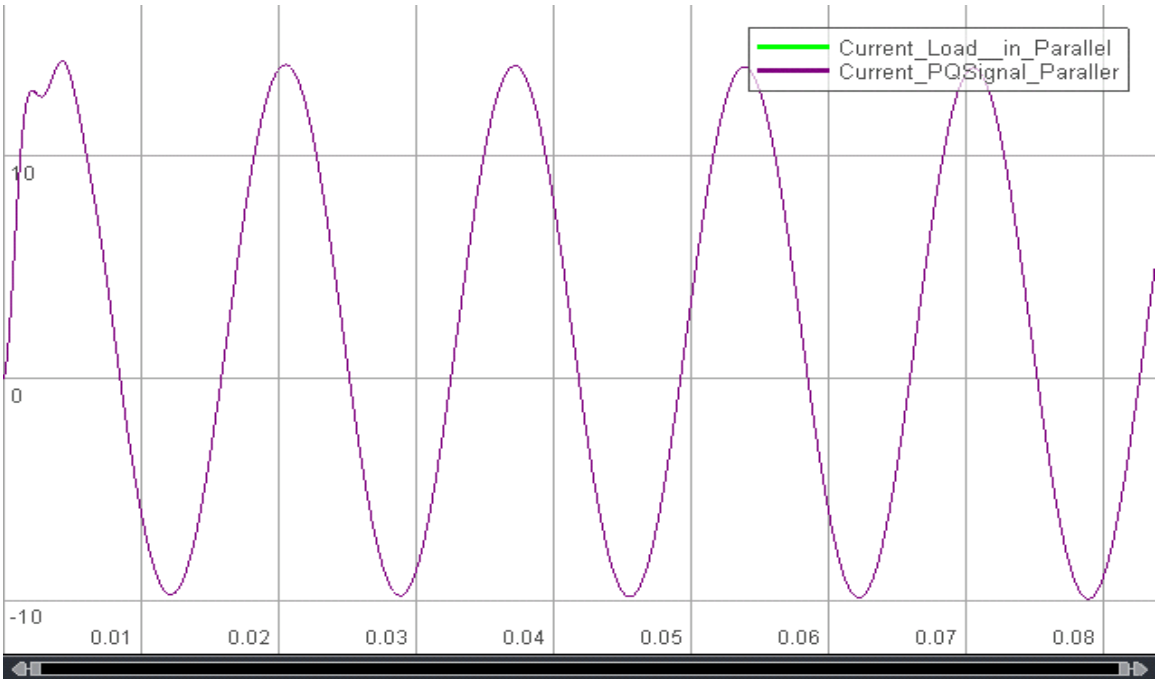


Figure 4.10 Current Waveforms for Controllable Load in Parallel Test

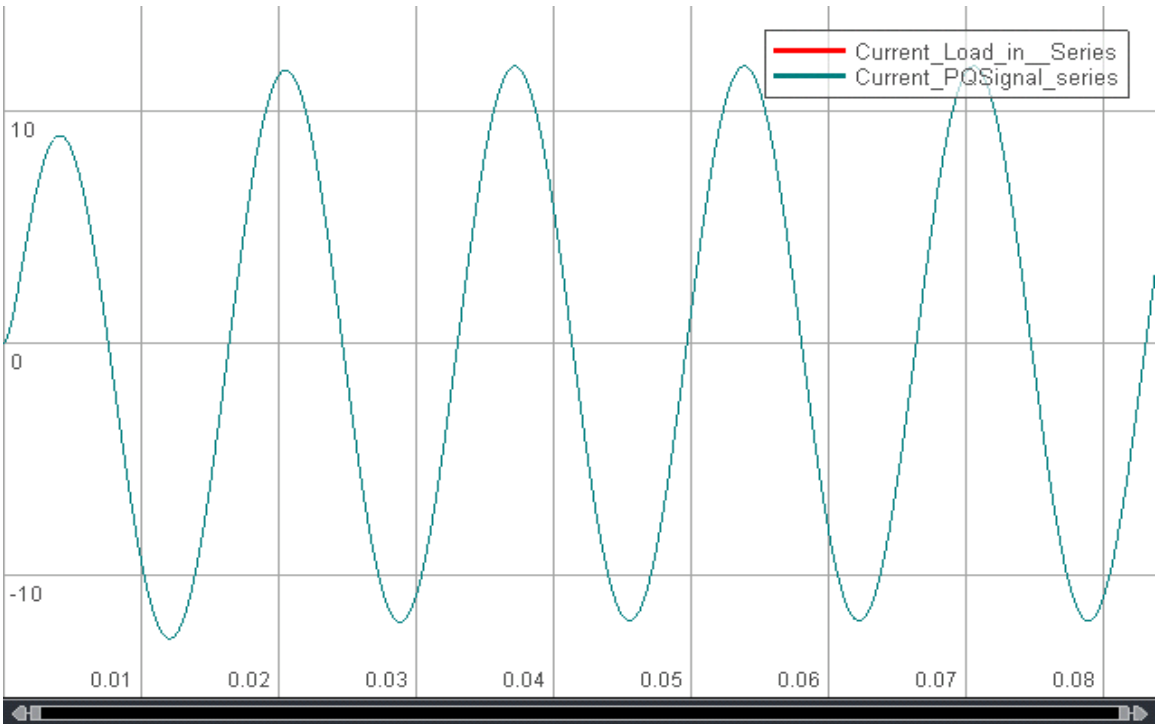


Figure 4.11 Current Waveforms for Controllable Load in Series Test

Observing the current waveforms in Figure 4.10 and 4.11 by zooming in shows an exact match. Thus, this match validatesthe controllable RLC-based load is validated.

For a polynomial load, a, b and c are changed to signal port data. Figure 4.12 ahwos the test circuit. Figure 4.13 shows the comparision of the signal controllable test results and the simulation without signal control. Table 4.3 shows the system specification.

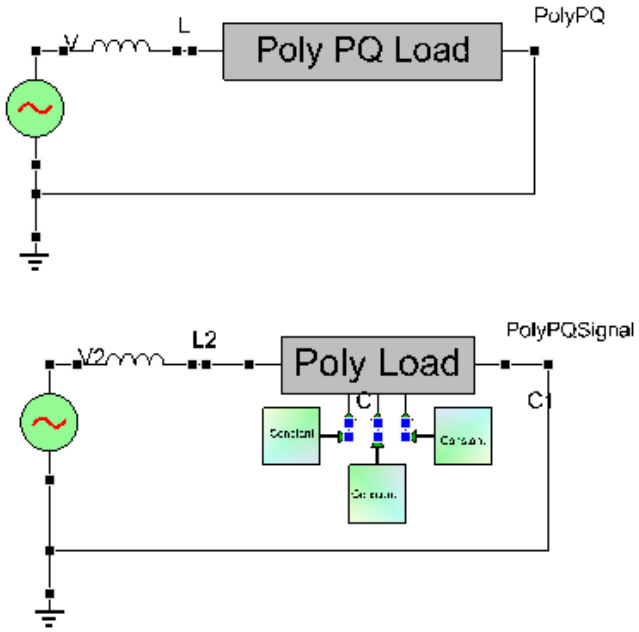


Figure 4.12 Controllable Polynomial PQ Load Model Test

Observing the current waveforms in Figure 4.13 by zooming in shows an exact match. Thus, thiss match validates the controllable polynomial load.

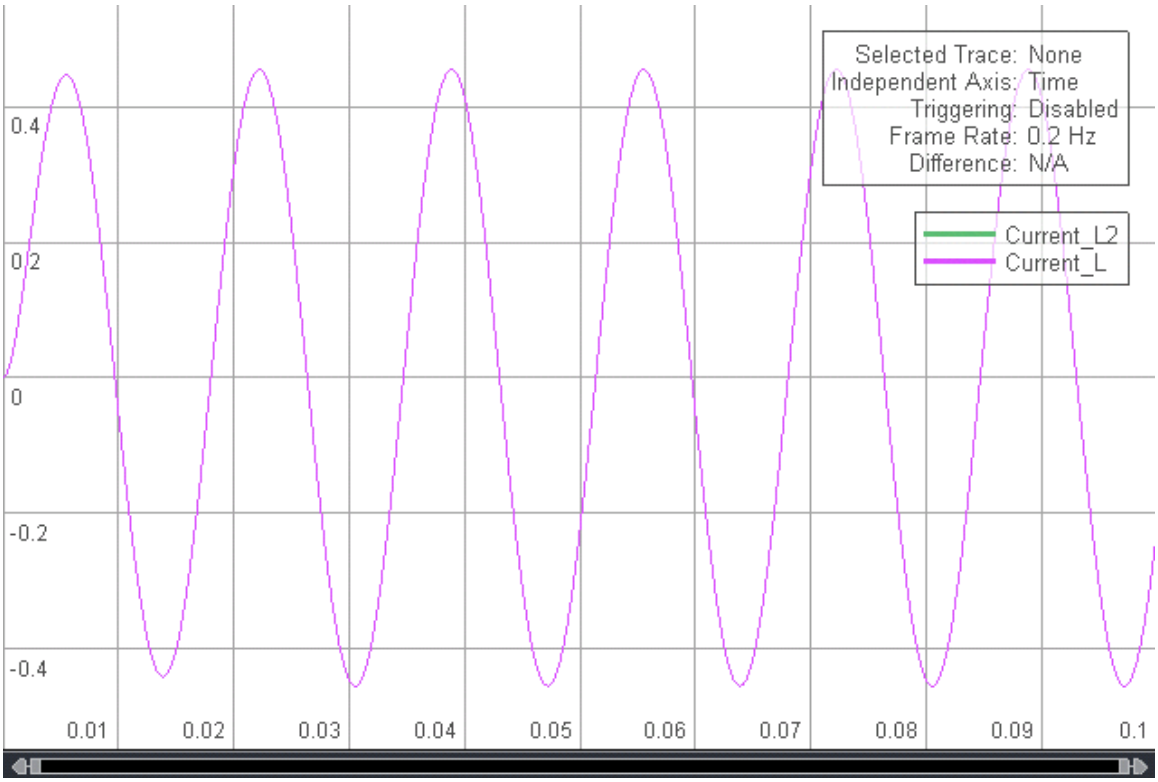


Figure 4.13 Current Waveforms for Controllable Polynomial Load Test

4.4 Results and discussion

The previous sections develop test cases are developed to test the different load models. Those load models have a steady state property. A polynomial load model for single-phase is implemented as a nonlinear load whose impedance changes according to the RMS measurement of node voltage. An RLC-based PQ load model based on constant impedance is presented and implemented as a linear load with a parallel or series combination of R, L and C elements. The comparision of of VTB simulation , PowerWorld simulation and Matlab/SimPowerSystem simulation validate the models.

The model test results in transient study match the outline of the solution. Since VTB uses a trapezoidal integration method, the solution is more sinusoidal and approaches the true solution.

The result of the steady state study matches the profile of the voltage, which is the state variable in VTB. A slight mismatch exists in the generator P and Q, caused by the power source modeling, where an ideal voltage source with a series resistance is used in VTB instead of a swing bus or PV buses. Fourier transform and truncation contributes to the mismatch in the calculation. Also, the power flow program tolerance and Jacobian formulation selection will affect the accuracy of solution.

The controllable load model gives exactly the same behavior as non-controllable load. The basic mathematical models for the controllable load model and the non-controllable load are the same. Only the control parameter comes from a different source, one from device parameters and the other from signal ports.

These new models provide a new tool for power system analysis. While VTB has been used frequently for power electronic applications, by extending its capabilities and models to the power system level, more analysis can be done to link the control, power electronic and power system aspects of operating a complex power grid. While initial efforts will focus on shipboard applications, these new models provide utility engineers with an additional tool to integrate various system responses to better understand transient and steady state response for the coupled terrestrial control and power system.

4.5 Summary

In this chapter, a polynomial load model and RLC based PQ model are derived and successfully implemented as a DLL Resistive Companion Form model in VTB. Powerworld, a power flow program, verified the validity of the load model in a steady state study. Matlab/Simulink simulation results verified the validity of the RLC-based PQ load models behavior in a time domain simulation. Then, those models, modified to be a controllable load by adding signal ports, were compared to a load model without control. Satisfactory results indicate that these models can be used in both steady state and transient studies. In the following chapter, these new load models will be used in the distributed simulation.

CHAPTER V

AGENT BASED DISTRIBUTED SIMULATION

This chapter mainly describes the agent based distributed algorithm. The natural coupling model and signal coupling models are developed and implemented in VTB. This chapter presents and analyze their performance in steady state and transient studies to verify the correctness and applicability of the proposed algorithm.

5.1 Introduction

Distributed simulation ussed agents to collect/send information to remote sites and receive/reproduce the information at local sites. Agents here can play a role in collecting measurements at the natural coupling level, as well as at the signal coupling level. Also, the agents can be programmed to reproduce the response to the local simulation with the environment.

Four kinds of agents are developed in the distributed simulation in this research and used at the natural coupling level and signal coupling level, respectively. Figure 5.1 shows heir symbols in VTB.

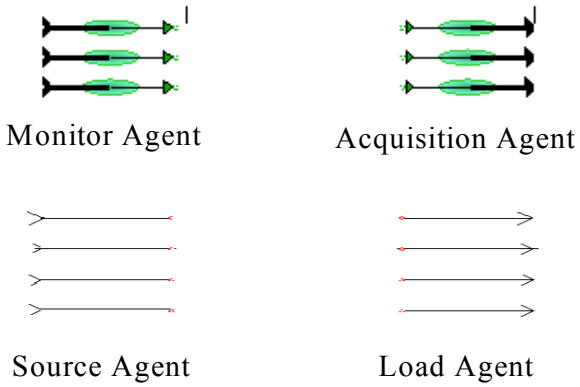


Figure 5.1 Symbols of Agents Models in VTB

The functions of the four agents are below:

1. Monitor agent

Used at signal coupling level. Will collect measurement and send signal command to remote site.

2. Acquisition agent

Used at signal coupling level. Will receive signal command from remote site and control component at local site.

3. Source agent

Used at natural coupling level. Will send boundary measurement to other agent and act as voltage source in local simulation.

4. Load agent

Used at natural coupling level. Will receive boundary measurement to other agent and act as load in local simulation.

5.2 Natural coupling model

The source agent and load agent are used at the natural coupling level. The decoupled algorithm discussed in the section below is embedded in agents to reproduce the correct response to the simulation. This section describes the algorithm extended from the DC coupled method to distributed simulation in the power system. The problem starts with the entire power system network. Suppose that two sub networks are connected by a tie line as Figure 5.2 shows:

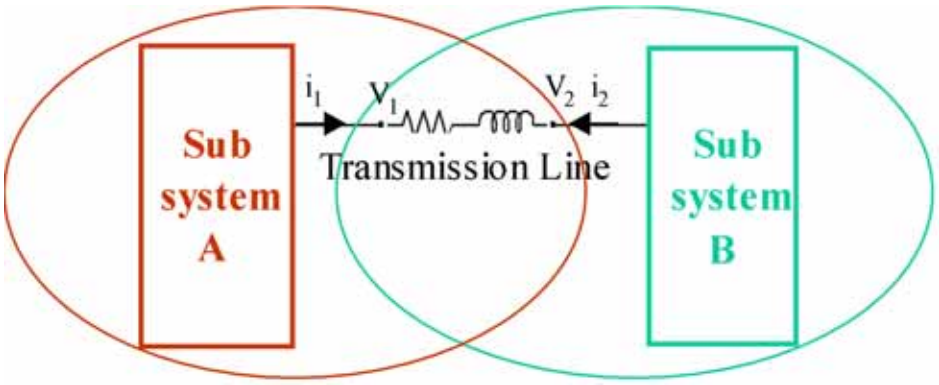


Figure 5.2 Whole System without Decoupling

The key issues for distributed simulation include decoupling the circuit and representing the missing subsystem. The choices of those two will affect the stability and accuracy of the solution.

5.2.1 Proposed workflow

Using the VI overlap decoupling method, the whole system can be decoupled into two subsystems with the transmission line present in both as the two circles indicate.

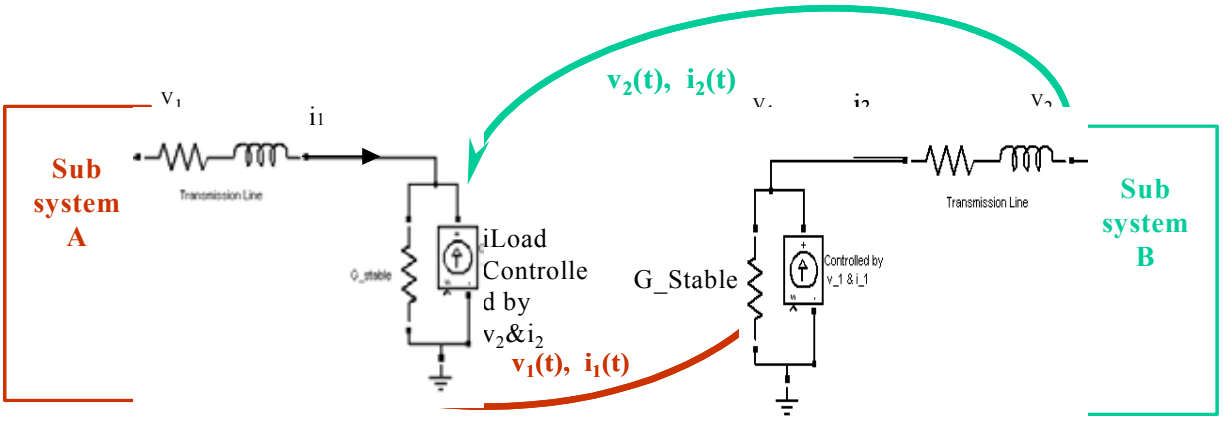


Figure 5.3 Whole System with Decoupling

When solving subsystem A, the subsystem B is treated as "missing system." A stabilizing resistor and a current source in parallel represent the missing subsystem, subsystem B in this case, as shown in left part of Figure 5.3. The corresponding point in the partner subsystem B controls their values. Similarly, when solving subsystem B, the subsystem A is treated as "missing system." A stabilizing resistor and a current source in parallel represent the missing subsystem, subsystem A in this case, as shown in right part of Figure 5.3.

Figure 5.4 shows the general workflow of the algorithm. For detailed implementation, if the inner loop runsonce, this algorithm is called linear method. For nonlinear method, if the stabilizing resistor is static, this algorithm is called non-adjusted stabilizing method. If the stabilizing resistor is varying, this algorithm is called adjusted stabilizing method.

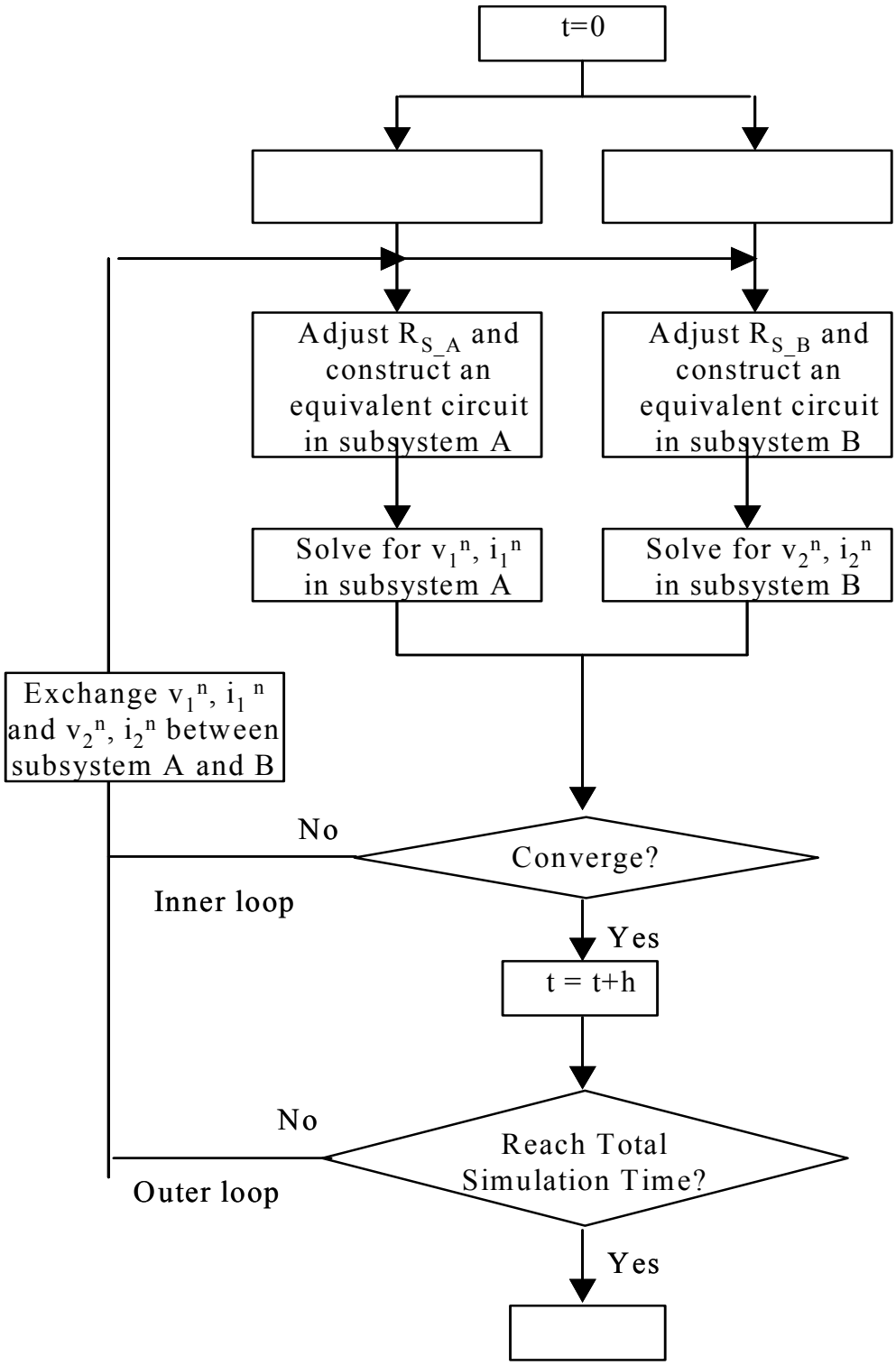


Figure 5.4 Workflow Of Distributed Simulation

The following are the detailed steps for solving subsystem A. In this workflow, the superscript n indicates the number of the inner iteration step.

1. Initialization: assume $v_2 = 0$ and $i_2 = 0$.

2. Construct equivalent circuit for subsystem B, set stabilizing resistor $R_{S\_A} = R_{S\_A}{}^0$, where $R_{S\_A}{}^0$ is a user defined value. Set the current source $i_{S\_A}{}^0 = \frac{v_2(t-h)}{R_{S\_A}} + i_2(t-h)$.

3. Solve for subsystem A and send $v_1{}^0$ and $i_1{}^0$ to subsystem B.

4. Receive $v_2{}^n$ and $i_2{}^n$ from subsystem B.

5. Construct equivalent circuit for subsystem B, for non-adjusted static stabilizing resistance, R_{S_A} stays static; for adjusted static stabilizing resistance R_{S_A} will be adjusted according to history data. c, $i_{S\_A}{}^n = \frac{v_2{}^n(t)}{R_{S\_A}} + i_2{}^n(t)$.

6. Solve for subsystem , get $v_1{}^{n+1}$ and $i_1{}^{n+1}$.

7. Check the convergence of i_1. If it converges, march to next time step. If it does not converge, send $v_1{}^{n+1}$ and $i_1{}^{n+1}$ to subsystem B and go to step 4.

Similarly this process goes through subsystem B:

1. Initialization: assume $v_1 = 0$ and $i_1 = 0$.

2. Construct equivalent circuit for subsystem B, set stabilizing resistor $R_{S\_B} = R_{S\_B}^{0}$, where $R_{S\_B}^{0}$ is a user defined value. Set the current source $i_{S\_B}^{0} = \frac{v_1(t-h)}{R_{S\_B}} + i_1(t-h)$.
3. Solve for subsystem B and send v_2^{0} and i_2^{0} to subsystem A.
4. Receive v_1^{n} and i_1^{n} from subsystem A.
5. Construct equivalent circuit for subsystem A, for non-adjusted static stabilizing resistance, RS_B will keep static; for adjusted static stabilizing resistance R_{S_B} will be adjusted according to history data. The next section presents detailed explanation about R_{S_B} selection. For the current source, $i_{S\_B}^{n} = \frac{v_1^{n}(t)}{R_{S\_B}} + i_1^{n}(t)$.
6. Solve for subsystem B, get v_2^{n+1} and i_2^{n+1}.
7. Check convergence of i_2. If converge, march to next time step. If not converge, send v_2 n+1 and i_2^{n+1} to subsystem A and go step 4.

At each time step, for a high accuracy simulation, the inner loop runs and stops when the current on the transmission line converges. For low accuracy or future hardware in the loop test, the inner loop runs only once, i.e. linear method. The outer loop runs to increase the simulation time until it reaches the total simulation time.

I developed agent models for single phase and three phases are developed in VTB to test the distributed simulation algorithm. Figure 5.5 shows those agent models' corresponding symbols. Those agents collect/send information to a remote site and

receive/reproduce the information at a local site. The agent models with arrow ends are used to send boundary measurements to others. The agent models with arrows are used to receive boundary measurements from other agents.

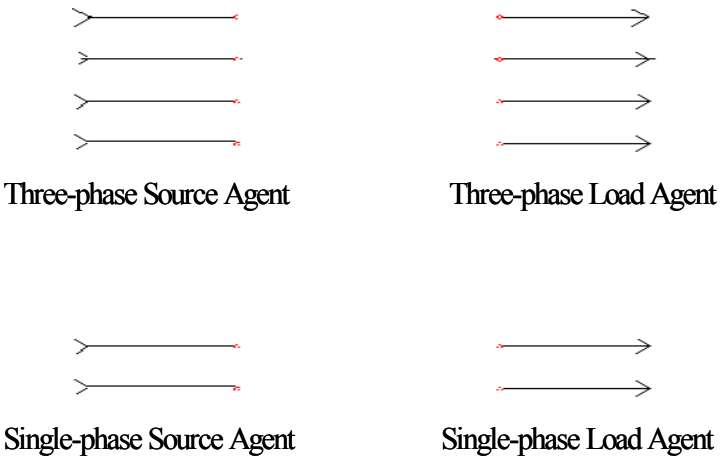
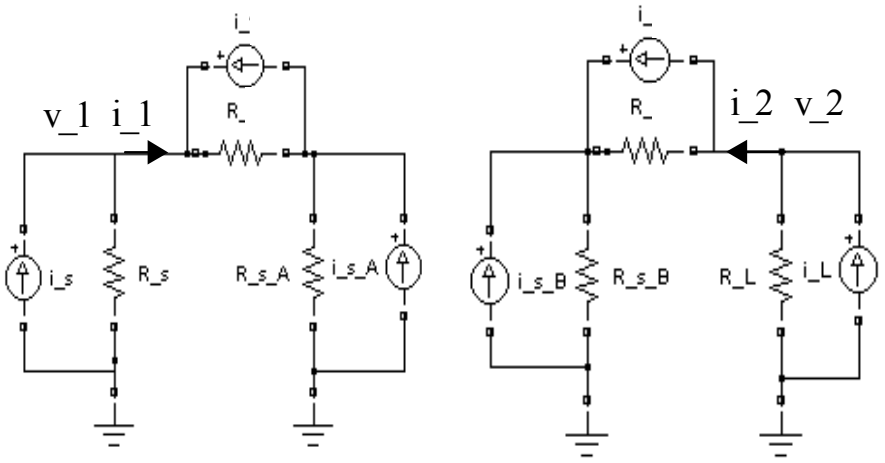


Figure 5.5 Agents Models in VTB

5.2.2 Numerical analysis

This section establishes the mathematical model of the distributed algorithm through circuit analysis and analyzes the convergence of different methods is. In each time step, after the original system is decoupled, the two subsystems are represented in the RCF model, as shown in Figure 5.6 below.



Subsystem A Subsystem B

Figure 5.6 RCF Equivalent Circuit for Subsystems

For subsystem A (left part of Figure.5.6), no matter how complex the known subsystem is, at each time step, simplified into a Norton equivalent and a current source i_S and a resistor R_S represent the subsystem A.. Following the RCF modeling, the transmission line is represented with a current source i_T and a resistor R_T. The missing subsystem B is represented by a current source i_{S_A} and a stabilizing resistor R_{S_A}.

Similarly, for subsystem B (right part of Figure.5.6), no matter how complex the known subsystem is, at each time step, simplified into a Norton equivalent and a current source i_L and a resistor R_L represent the subsystem B. Following the RCF modeling, the transmission line is represented with i_T and R. The missing subsystem A is represented by a current source i_{S_B} and a stabilizing resistor R_{S_B}.

According to the algorithm described in Figure 5.4, the relationship between the voltage and current of each iteration in the inner loop can be determined through nodal

analysis. According to the algorithm, v_2^n and i_2^n controls the value of i_{S_A}, and v_1^n and i_1^n controls the valu e of i_{S_B}. All other elements like resistors and current source are known and fixed at this time step. Thus, v_2^{n+1} and i_2^{n+1} is a function of v_1^n and i_1^n. Similarly, v_1^{n+1} and i_1^{n+1} is a function of v_2^n and i_2^n. The section below gives the detailed derivation.

For subsystem A, the power source and transmission line until node 2 can be simplified into a Norton equivalent. The equivalent current source and resistance can be found through the open circuit and closed circuit analysis as shown below:

Open circuit at node 2:

$v_{open}(t)=i_s(t)*R_s$

Close circuit at node 2:

$$i_{close}(t)=-(i_s(t)+i_T(t))*\frac{R_T}{R_s+R_T}+i_s(t)=\frac{R_S}{R_s+R_T}i_s(t)-\frac{R_T}{R_s+R_T}i_T(t)$$

$$i_{eq\_1}(t)=i_{close}(t) \tag{5.1}$$

$$R_{eq\_1}=\frac{v_{open}}{i_{close}}=\frac{1}{\frac{1}{R_s+R_T}-\frac{1}{R_s+R_T}\frac{R_T}{R_s}\frac{i_T(t)}{i_s(t)}} \tag{5.2}$$

Simplified circuit:

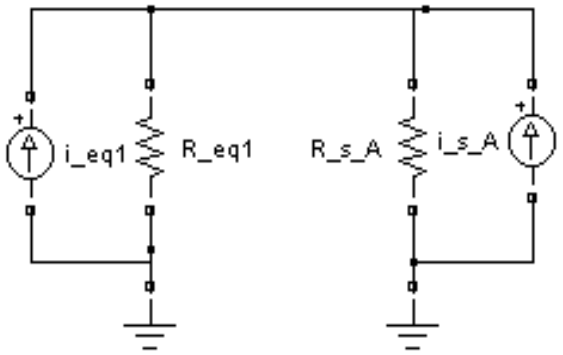


Figure 5.7 Simplified Circuit with Norton Equivalent of Subsystem A

From Figure 5.7, $i_1^{n+1}(t)$ can be expressed as (5.3):

$$i_1^{n+1}(t) = i_{eq\_1}(t) - (i_{eq\_1}(t) + i_{s\_A}^{n}(t)) \frac{R_{s\_A}}{R_{sSA} + R_{eq\_1}}$$

$$= \frac{R_{eq\_1}}{R_{S\_A} + R_{eq\_1}} i_{eq\_1}(t) - \frac{R_{s\_A}}{R_{S\_A} + R_{eq\_1}} i_{SS\_A}^{n}(t) \quad (5.3)$$

From Figure 5.6, $v_1^{n+1}(t)$ can be expressed as (5.4):

$$v_1^{n+1}(t) = R_s(i_s(t) - i_1^{n+1}(t)) \quad (5.4)$$

Here, only $v_1(t)$ and $i_1(t)$ are unknowns that need to be determined through iterations. i_{s_a} is decided by $v_2(t)$, $i_2(t)$ and R_{S_A}. Other variables such as $i_{eq\_1}(t)$, $i_s(t)$ and all the resistance are fixed at this point.

Following the same process for subsystem A, subsystem B can be simplified. Here the power load and transmission line until node 1 will be simplified into Norton equivalent. The open circuit and closed circuit analysis determines the equivalent current source and resistance as shown below:

Open circuit at node 2:

$v_{open}(t) = i_L(t)*R_L$

Close circuit at node 2:

$$i_{close}(t) = -(i_T(t) - i_L(t))*\frac{R_L}{R_L+R_T} + i_T(t) = \frac{R_L}{R_L+R_T} i_L(t) + \frac{R_T}{R_L+R_T} i_T(t)$$

$i_{eq\_2}(t) = i_{close}(t)$

$$R_{eq\_2} = \frac{v_{open}}{i_{close}} = \frac{1}{\frac{1}{R_L+R_T} + \frac{1}{R_L+R_T}\frac{R_T}{R_L}\frac{i_T(t)}{i_L(t)}}$$

Simplified circuit:

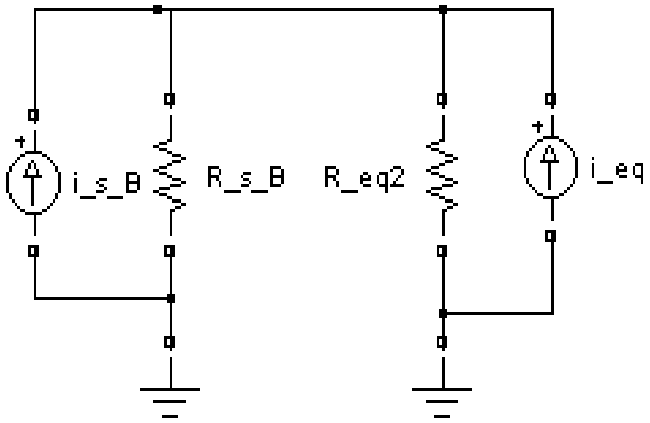


Figure 5.8 Simplified Circuit with Norton Equivalent of Subsystem B

From Figure 5.8, $i_2^{n+1}(t)$ can be expressed as (5.5):

$$i_2^{n+1}(t) = i_{eq\_2}(t) - (i_{eq\_2}(t) + i_{s\_B}^{n}(t))\frac{R_{s\_B}}{R_{s\_B}+R_{eq\_2}}$$

$$= \frac{R_{eq\_2}}{R_{S\_B}+R_{eq\_2}} i_{eq\_2}(t) - \frac{R_{Ss\_B}}{R_{S\_B}+R_{eq\_2}} i_{S\_B}^{n}(t) \qquad (5.5)$$

From Figure 5.6, $v_1^{n+1}(t)$ can be expressed as (5.6):

$$v_2^{n+1}(t) = R_L(i_L(t) - i_2^{n+1}(t)) \tag{5.6}$$

Here, only $v_2(t)$ and $i_2(t)$ are unknowns that must be determined through iterations. i_{s_a} is decided by $v_1(t)$, $i_1(t)$ and R_{S_B}; other variables such as $i_{eq\_2}(t)$, $i_L(t)$ and all the resistance are fixed at this point.

Combining equations (5.3)-(5.6)

$$i_1^{n+1}(t) = \frac{R_{eq\_1}}{R_{Ss\_A} + R_{eq\_1}} i_{eq\_1}(t) - \frac{R_{S\_A}}{R_{S\_A} + R_{eq\_1}} i_{S\_A}^n(t) \tag{5.7}$$

$$i_2^{n+1}(t) = \frac{R_{eq\_2}}{R_{S\_B} + R_{eq\_2}} i_{eq\_2}(t) - \frac{R_{s\_B}}{R_{S\_B} + R_{eq\_2}} i_{S\_B}^n(t) \tag{5.8}$$

Here,

$$i_{s\_A}^n(t) = \frac{v_2^n(t)}{R_{S\_A}} + i_2^n(t)$$

$$= -\frac{R_L(i_2^n(t) - i_L(t))}{R_{S\_A}} + i_2^n(t) \quad (\text{from } v_2^{n+1}(t) = R_L(i_L(t) - i_2^{n+1}(t)))$$

$$= (1 - \frac{R_L}{R_{S\_A}}) i_2^n(t) + \frac{R_L}{R_{S\_A}} i_L(t)$$

$$i_{s\_B}^n(t) = \frac{v_1^n(t)}{R_{S\_B}} + i_1^n(t)$$

$$= -\frac{R_S(i_1^n(t) - i_s(t))}{R_{S\_B}} + i_1^n(t) \quad (\text{from } v_1^{n+1}(t) = R_s(i_s(t) - i_1^{n+1}(t)))$$

$$= (1 - \frac{R_S}{R_{S\_B}}) i_1^n(t) + \frac{R_S}{R_{S\_B}} i_s(t)$$

Replace the $i_{S\_A}^n(t)$ and $i_{S\_B}^n(t)$ in equation (5.7) and (5.8)

$$i_1^{n+1}(t) = \frac{R_{eq\_1}}{R_{S\_A} + R_{eq\_1}} i_{eq\_1}(t) - \frac{R_{s\_A}}{R_{S\_A} + R_{eq\_1}} \left(\left(1 - \frac{R_L}{R_{S\_A}}\right) i_2^n(t) + \frac{R_L}{R_{S\_A}} i_L(t) \right)$$

$$i_2^{n+1}(t) = \frac{R_{eq\_2}}{R_{S\_B} + R_{eq\_2}} i_{eq\_2}(t) - \frac{R_{s\_B}}{R_{S\_B} + R_{eq\_2}} \left(\left(1 - \frac{R_S}{R_{S\_B}}\right) i_1^n(t) + \frac{R_S}{R_{S\_B}} i_s(t) \right)$$

Decoupling the changing part from the fix part

$$i_1^{n+1}(t) = \frac{R_L - R_{s\_A}}{R_{S\_A} + R_{eq\_1}} i_2^n(t) - \frac{R_L}{R_{S\_A} + R_{eq\_1}} i_L(t) + \frac{R_{eq\_1}}{R_{S\_A} + R_{eq\_1}} i_{eq\_1}(t)$$

$$i_2^{n+1}(t) = \frac{R_S - R_{s\_B}}{R_{S\_B} + R_{eq\_2}} i_1^n(t) - \frac{R_S}{R_{S\_B} + R_{eq\_2}} i_S(t) + \frac{R_{eq\_2}}{R_{S\_B} + R_{eq\_2}} i_{eq\_2}(t)$$

Let

$$J_1 = -\frac{R_L}{R_{s\_A} + R_{eq\_1}} i_L(t) + \frac{R_{eq\_1}}{R_{s\_A} + R_{eq\_1}} i_{eq\_1}(t)$$

$$J_2 = -\frac{R_S}{R_{s\_B} + R_{eq\_2}} i_S(t) + \frac{R_{eq\_2}}{R_{s\_B} + R_{eq\_2}} i_{eq\_2}(t)$$

The iteration matrix is expressed in the equation (5.9)

$$\begin{bmatrix} i_1^{n+1}(t) \\ i_2^{n+1}(t) \end{bmatrix} = \begin{bmatrix} 0 & \frac{R_L - R_{S\_A}}{R_{eq\_1} + R_{S\_A}} \\ \frac{R_S - R_{S\_B}}{R_{eq\_2} + R_{S\_B}} & 0 \end{bmatrix} \begin{bmatrix} i_1^n(t) \\ i_2^n(t) \end{bmatrix} + \begin{bmatrix} J_1 \\ J_2 \end{bmatrix} \tag{5.9}$$

Also, from the description of the algorithm, the initial current is expressed in the equation (5.10) and (5.11) below. Assume the current from the last step sets up for the equivalent current source.

$$i_1^0(t) = \frac{R_{eq\_1}}{R_{S\_A} + R_{eq\_1}} i_{eq\_1}(t) - \frac{R_{s\_A}}{R_{S\_A} + R_{eq\_1}} i_{S\_A}^0 \quad (5.10)$$

$$i_2^0(t) = \frac{R_{eq\_2}}{R_{S\_B} + R_{eq\_2}} i_{eq\_2}(t) - \frac{R_{s\_B}}{R_{Ss\_B} + R_{eq\_2}} i_{S\_B}^0 \quad (5.11)$$

$$i_{S\_A}{}^0 = \frac{v_2(t-h)}{R_{S\_A}} + i_2(t-h)$$

$$i_{S\_B}{}^0 = \frac{v_1(t-h)}{R_{S\_B}} + i_1(t-h)$$

Here, h is the time step size.

$$R_L = \frac{2L_{load}}{h} + R_{load}$$

$$R_T = \frac{2L_{trans\_line}}{h} + R_{trans\_line}$$

$$R_{eq\_1} = \frac{1}{\frac{1}{R_s + R_T} - \frac{R_T i_T(t)}{(R_s + R_T) R_s i_s(t)}}$$

$$R_{eq\_2} = \frac{1}{\frac{1}{R_L + R_T} + \frac{R_T i_T(t)}{(R_L + R_T) R_L i_L(t)}}$$

$$i_L(t) = -\frac{1}{\frac{2L_{load}}{h} + R_{load}} v_2(t-h) - \frac{\frac{2L_{load}}{h} - R_{load}}{\frac{2L_{load}}{h} + R_{load}} i_1(t-h)$$

$$i_T(t) = -\frac{1}{\frac{2L_{trans\_line}}{h} + R_{trans\_line}} v_2(t-h) - \frac{\frac{2L_{trans\_line}}{h} - R_{trans\_line}}{\frac{2L_{trans\_line}}{h} + R_{trans\_line}} i_1(t-h)$$

$$i_{eq\_1}(t) = \frac{R_S}{R_s + R_T} i_s(t) - \frac{R_T}{R_s + R_T} i_T(t)$$

$$i_{eq\_2}(t) = \frac{R_L}{R_L + R_T} = \frac{R_L}{R_L + R_T} i_L(t) + \frac{R_T}{R_L + R_T} i_T(t)$$

For the linear VI coupling model, $i_1^0(t)$ and $i_2^0(t)$ are the solutions for that time step. This calculation works for linear VI coupling model too. Since no inner loop for linear VI coupling exists, 5.10 and 5.11 turns out to be the final iteration matrix for the linear method.

For the non-linear method, to make the solution stable and converging, the eigenvalue of the iteration matrix must be within the unit circle. However, the eigenvalue, λ, $\frac{R_L - R_{S\_A}}{R_{eq\_1} + R_{S\_A}}$ and $\frac{R_S - R_{S\_B}}{R_{eq\_2} + R_{S\_B}}$, are state variables related and sensitive to time step size. To make eigenvalues $|\lambda|$ within 1, the best choice is to approximate $R_{S\_A} = R_L$ and $R_{S\_B} = R_S$.

But, as stated in the algorithm, only boundary voltage and current measurements are available for the unknown subsystem, R_L and R_S are unknown. The following relationship following between voltage and current in iterations allows the estimation of R_L and R_S from the measurement:

$$i_s^{\,n} = \frac{v_1^{\,n}}{R_s} + i_1^n = \frac{v_1^{\,n-1}}{R_s} + i_1^{n-1} \tag{5.12}$$

$$i_L^{\,n} = \frac{v_2^{\,n}}{R_L} + i_2^n = \frac{v_2^{\,n-1}}{R_L} + i_2^{n-1} \tag{5.13}$$

Note that i_s^n and i_L^n are fixed at a specific time step in equitation (5.12) and (5.13). Thus, R_L and R_S can be calculated through the boundary voltage and current measurement. Set $R_{S\_A} = R_L$ and $R_{S\_B} = R_S$; and derived the following equation for the estimation of R_{S_A} and R_{S_B}:

$$R_{s\_A} = -\frac{v_2^n - v_2^{n-1}}{i_2^n - i_2^{n-1}} \tag{5.14}$$

$$R_{S\_B} = -\frac{v_1^n - v_1^{n-1}}{i_1^n - i_1^{n-1}} \tag{5.15}$$

With such a stabilizing resistance selection, the solution is guaranteed to converge.

From Figure 5.5 and 5.6, the true solution is as follows:

$$i_1(t) = -\frac{R_L}{R_L + R_{eq\_1}} i_L(t) + \frac{R_{eq\_1}}{R_L + R_{eq\_1}} i_{eq\_1}(t) \tag{5.16}$$

$$i_2(t) = -\frac{R_S}{R_S + R_{eq\_2}} i_S(t) + \frac{R_{eq\_2}}{R_S + R_{eq\_2}} i_{eq\_2}(t) \tag{5.17}$$

Here, i_1 and i_2 are of the same magnitude and are opposite of each other.

Observing equation (5.9), with the selection of R_{S_A} and R_{S_B} as indicated by (5.14) and (5.15), $i_2^{n+1}(t)$ and $i_2^{n+1}(t)$ will converge to J_1 and J_2, which are equivalent with (5.16) and (5.17). Therefore, the adjusted stabilizing resistance method will guarantee that the solution converges to the true solution.

5.2.3 *Algorithm Test*

To test the proposed three phase decoupling method, I developed a program in Matlab, where I could concentrate on the algorithm without really programming with the RPC communications.

To imitate the process, I created a program to mimic communications between simulations and the distributed simulations share results at the coupling point only. Figure 5.9 shows the architecture.

The boundary values are passed between subsystems as function arguments. In each subsystem, the global variant only includes the subsystem information. The time domain RCF is used to develop each device model, nodal analysis solves each subsystem. The simple test case from the MURI tested the algorithms.

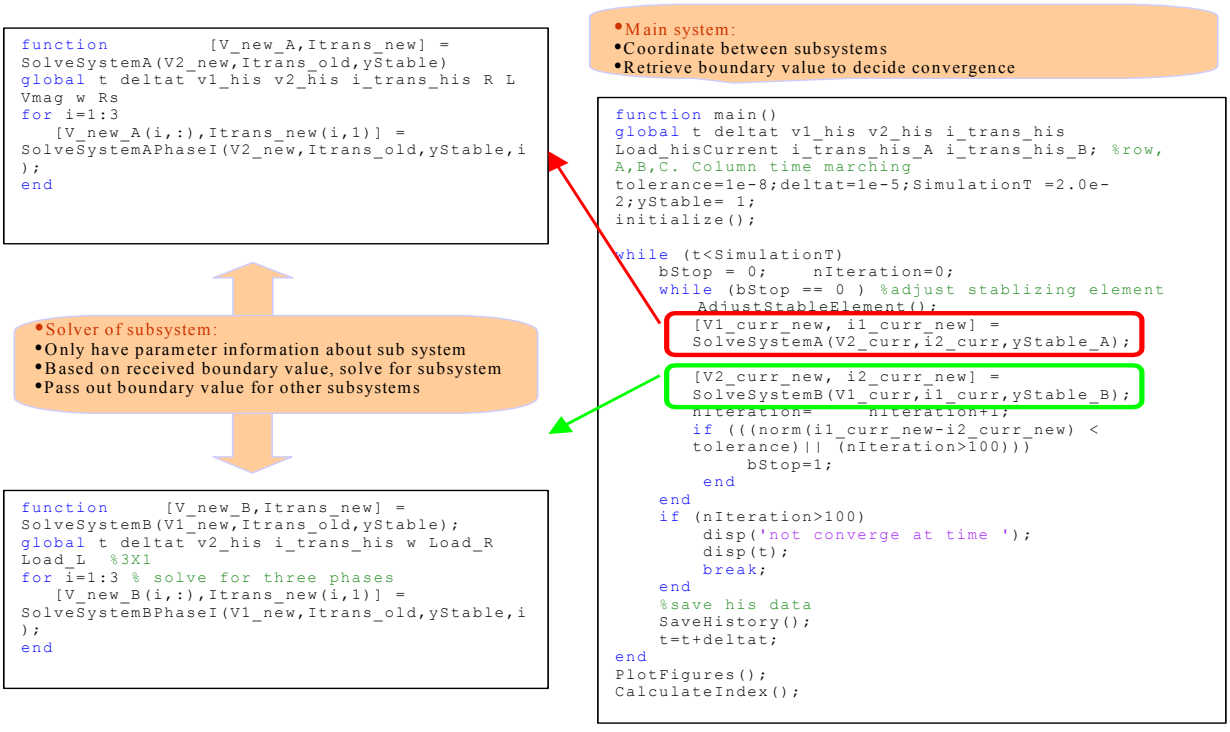


Figure 5.9 Program Architecture for Distributed Simulation

A. Single source and single load

This algorithm was first tested with a power system shown in Figure 5.10 below:

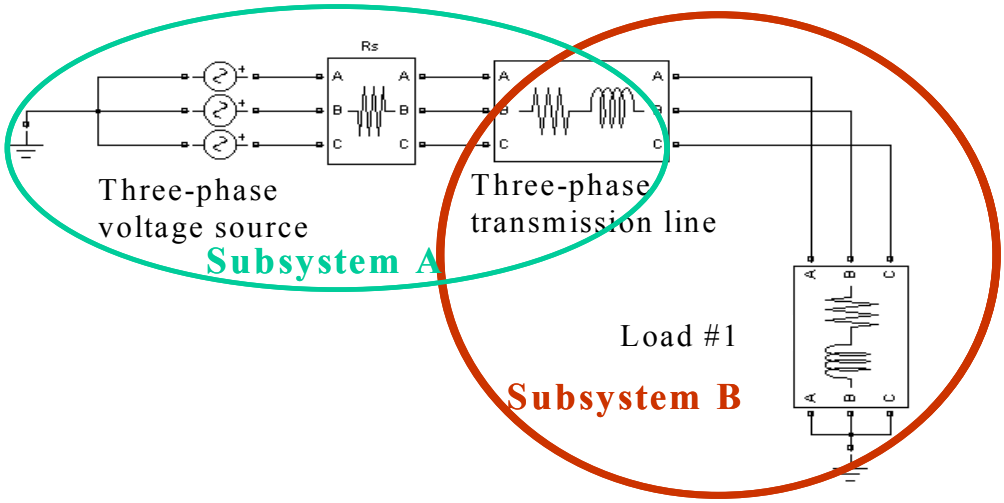


Figure 5.10 Distributed Simulation Test Case #1 Diagram

The system specification comes from the MURI project as shown in Table 5.1.

Table 5.1

System Specification for Distributed Simulation Test Case #1

Voltage source	RMS:120 V Rs=2 Ohms
Load #1	R = 12.3Ω, L = 0.03138H
Transmission Line	R = 0.34Ω, L = 0.005913H

The whole system is decoupled through the transmission line and solved by the Matlab program. To validate the solution from distributed simulation, this test case was also simulated in Simulink using the power system block set with the same specifications. Figure 5.11 shows the current flow through the transmission line from the distributed

simulation and non-distributed simulation. The average deviation between distributed simulation and non-distributed simulation is 0.173%.

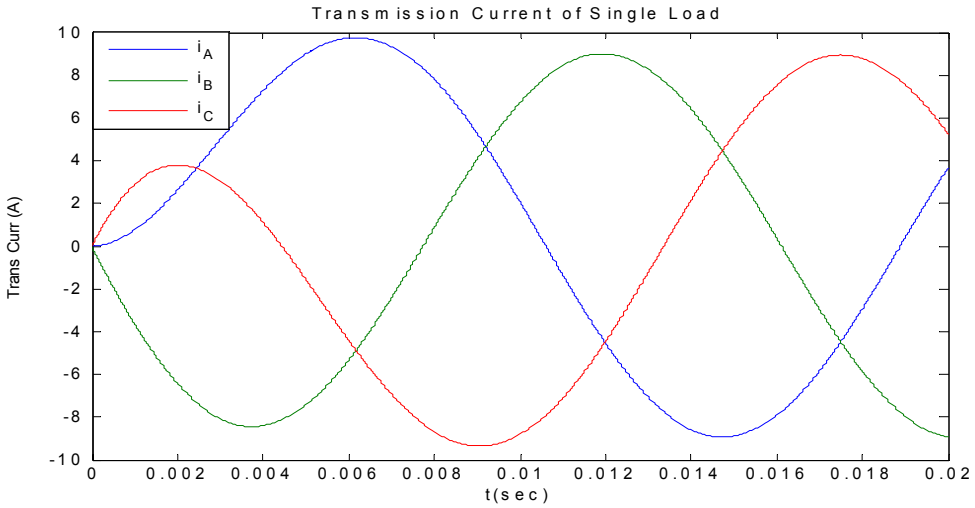


Figure 5.11 Current Waveform from Distributed Simulation and Non-Distributed of Test Case #1

B. Single source and multiple loads

This algorithm is then tested with a power system shown in Figure 5.12.

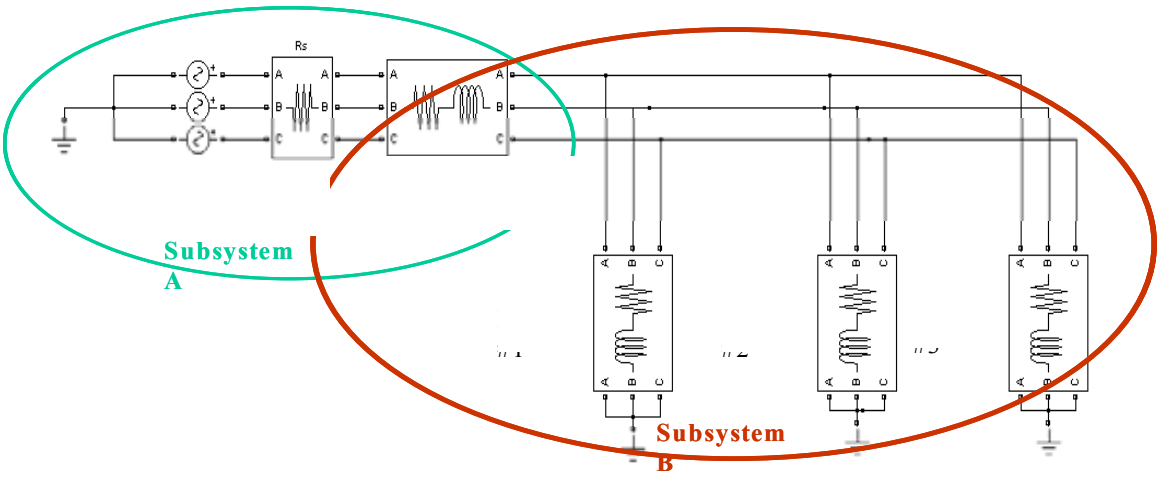


Figure 5.12 Distributed Simulation Test Case #2 Diagram

The system specification was listed in Table 5.2 as follows:

Table 5.2

System Specifications for Test Case #2 [1]

Voltage source	RMS:120 V, Rs=2 Ohms
Load #1	R = 12.3Ω, L = 0.03138 H
Load #2	R = 14.52Ω, L = 0.031468 H
Load #3	R = 14.52Ω, L = 0.031468 H
Transmission Line	R = 0.34Ω, L = 0.005913 H

The whole system was decoupled through the transmission line and solved by the Matlab program. The current flows through the transmission line from the distributed simulation and non-distributed simulation are shown in Figures 5.12. The average deviation between distributed simulation and non-distributed simulation is 0.4475%.

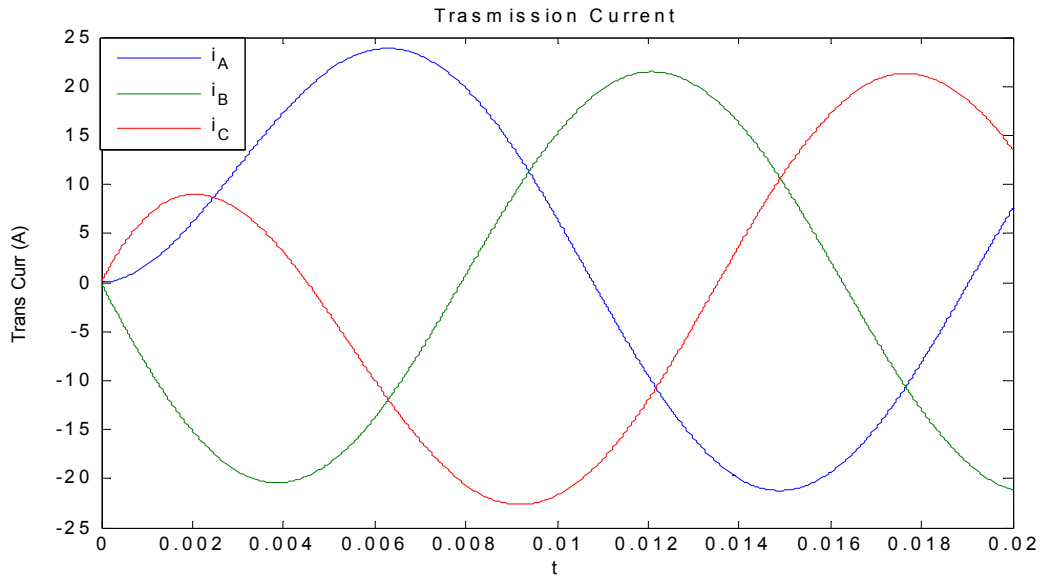


Figure 5.13 Current Waveform from Distributed Simulation and Non-Distributed of Test Case #2

These two test cases show clearly that the three-phase VI decoupling method works for time domain simulation, and achieves satisfactory results. Also, for static stabilizing element algorithm the average number of iterations to reach convergence is six. This number will be larger when the network gets more complex and multiple coupling points are present. For the adjustable stabilizing element algorithm, the average number of iterations to reach convergence is two. This number would be the same even when the network gets more complex and multiple coupling points are presente.

This section proposes a general three-phase VI coupling method for distributed simulation. This algorithm is tested with two simple cases, and desired results are achieved. However, the solver constructed in a Matlab program is a simple one as it is designed for the RL load and can only deal with simple networks. VTB has many of models and provides a solver with optimization. Therefore, in the next step, we will migrate the algorithm into the VTB framework and test it with different system configurations.

5.3 Signal coupling model

The monitor agent and acquisition agent are used at the signal coupling level. For most controller devices, especially digital controllers, a digital signal processor (DSP) is used. A DSP acquires measurement from the power system and performs data analysis to reach a decision for the next step. Thus, the signal of command is based on the history data and only the measurement at the current step affects the signal of next step.

Therefore, considering the naturel of the signal, signal coupling can be realized by direct feed and through method. Here, the only limitation is that the time step cannot be larger than the controller sample time, or it may greatly affected the accuracy.

In VTB implementation, the signal coupling models, the monitor agent and acquisition agent are implemented as linear models. The acquisition agent accepts a signal from any source and passes it out to subscribed monitor agents through RPC communication over network. The monitor agent agents can select the acquisition agent to subscribe data. Both agents can be scalable on the port number. Time step is the only factor that influences the solution accuracy.

5.4 Test cases and performance analysis

To demonstrate the distributed simulation algorithm, corresponding models are developed in VTB using C++ language and are tested with different network configurations. The models performance is analyzed in time domain and steady state simulation.

5.4.1 Two-bus system with natural coupling

This section presents a comparision of the performance of several different algorithms including including linear method, static stabilizing resistance and adjustable stabilizing resistance method based on a two-bus system. Figure 5.14 shows the two-bus system, and Table 5.2 shows the system specification from the MURI project.

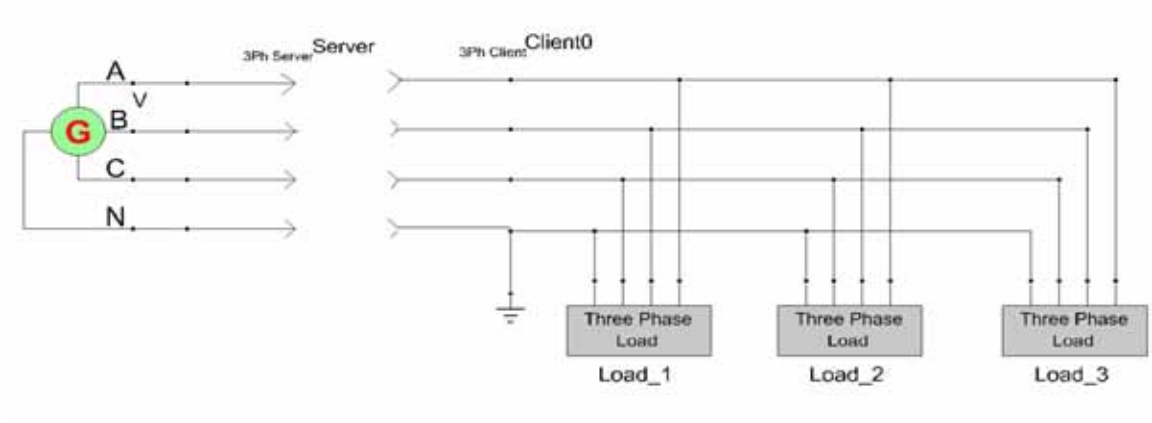


Figure 5.14 Two-bus System in VTB for Distributed Simulation

Chapter 4 contains the developemnt of the PQ load and signal-controlled PQ load in the test case; comparison of the load model with Matlab/Simulink test has validated them.

Distributed simulation is carried out for a different connected number of loads with different step sizes. The adjusted stabilizing resistance algorithm is compared with non-adjusted stabilizing resistance and the linear algorithm. The results of all distributed test cases are compared with a single simulation with the same time step to evaluate the performance of distributed simulation. Its performance is measured by the Mean Absolute Percentage Error (MAPE) in the transmission line current as shown in equation (5.19). Table 5.3 shows the test results.

$$APE = \frac{|\, i_{non-distributed} - i_{distribued} \,|}{i_{non-distributed}} * 100\% \tag{5.18}$$

$$MAPE = \frac{1}{N_h} \sum_{N_h} APE \tag{5.19}$$

Here, N is the number of sample points in the simulation.

Table 5.3

MAPE for Distributed Simulation

Case	Step size (s)	Linear	Non-Adjusted Stabilizing resistance	Adjusted Stabilizing resistance
Single Load	1e-6	0.332%	0.0248%	0.0248%
	1e-5	2.741%	0.0248%	0.0248%
	1e-4	15.733%	Did not converge	0.0248%
	1e-3	422.338%	Did not converge	0.0248%
Three Load	1e-6	0.3212%	0.0323%	0.0323%
	1e-5	3.7.74%	0.0928%	0.0928%

This comparison shows quicker convergence and higher accuracy for the adjusted stabling resistance algorithm. In each time step, the adjusted stabling resistance method can find the solution within tolerance in two iterations while the non-adjusted stabilizing resistance method needs three to four iterations to reach the solution for converged cases.

Fewer iterations lead to less computation time. The adjusted stabling resistance method always converges and is not time step sensitive. For the linear method, the MAPE keeps growing exponentially when the time step size increases, because the larger the step size, the less accuracy of the discretized RCF model for the components in simulation. Without the correction in the minor step for the non-linear method, the error accumulates and finally makes the simulation results completely inaccurate. The linear method can have better performance when the correction based on the previous calculation is introduced for next step simulation using a state estimator. The accuracy

decreases when the number of components in the simulation increases, for all three methods, because the VTB solver only has control over the voltage profile, while the currents are calculated based on an individual device model; thus, the truncation error accumulates.

5.4.2 Two-bus system with signal coupling

To demonstrate the agent based distributed simulation in signal coupling area, this section presents two simple test cases.

This first test case in Figure 5.15 is used to demonstrate the agent model working with pure signal level coupling. Figure 5.16 shows the simulation results of the transmission line current. The results, compared with a single simulation, shows the performance of distributed simulation with diffeerent methods. The system specification comes from the MURI project, as shown in Table 5.1.

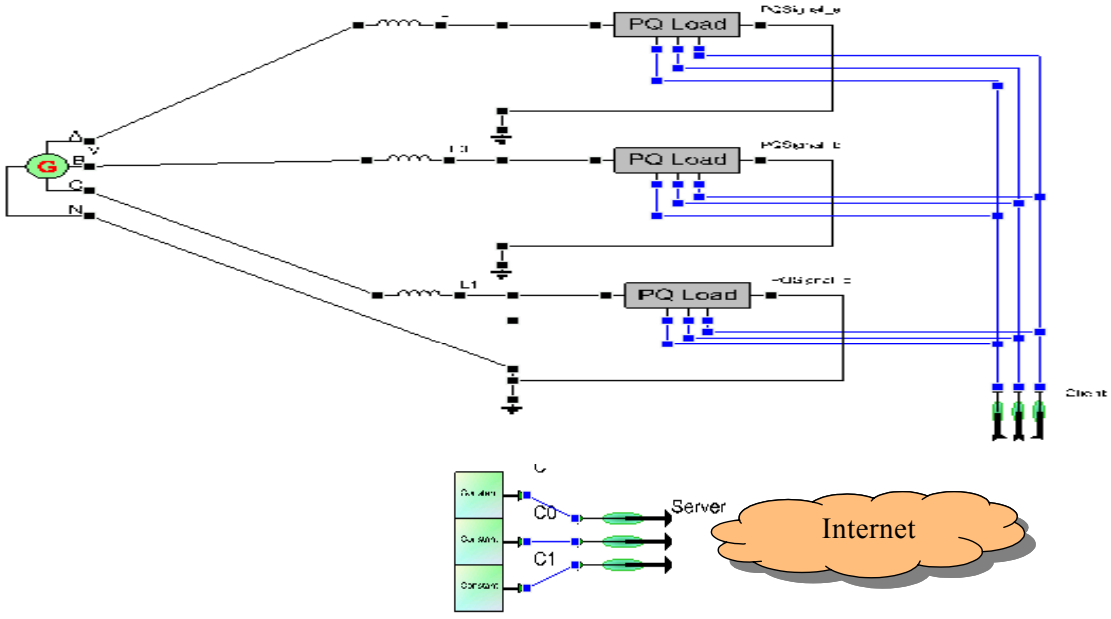


Figure 5.15 Signal Coupling Test in VTB with Constant Signal Source

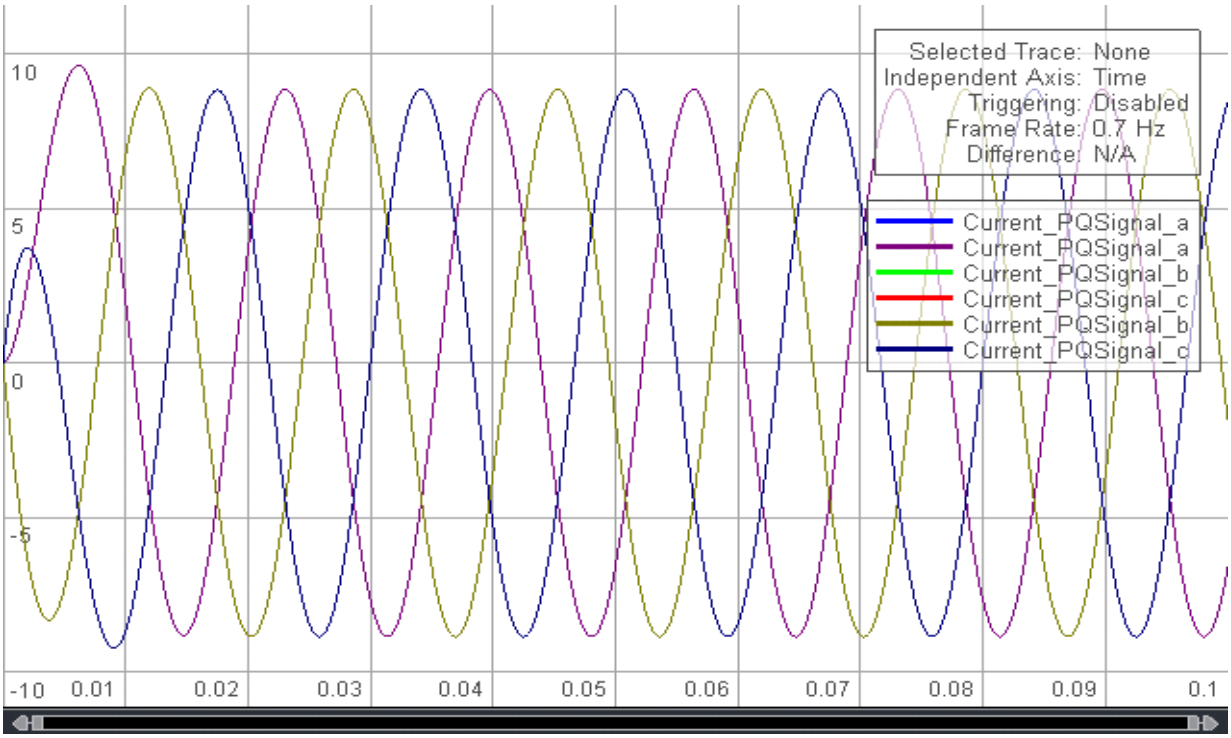


Figure 5.16 Signal Coupling Test Result in VTB with Constant Signal Source

For this simple system with constant signal source, the waveforms of the three-phase transmission line from distributed simulations match the waveforms from non-distributed simulation exactly, as observed in Figure 5.16.

The second test case shown in Figure 5.17, tests the influence of changing signal in distributed simulation. The load control signal is a step signal from a remote site. Figure 5.18 shows the simulation result of the transmission line current. The results are compared with a single simulation to see the performance of distributed simulation.

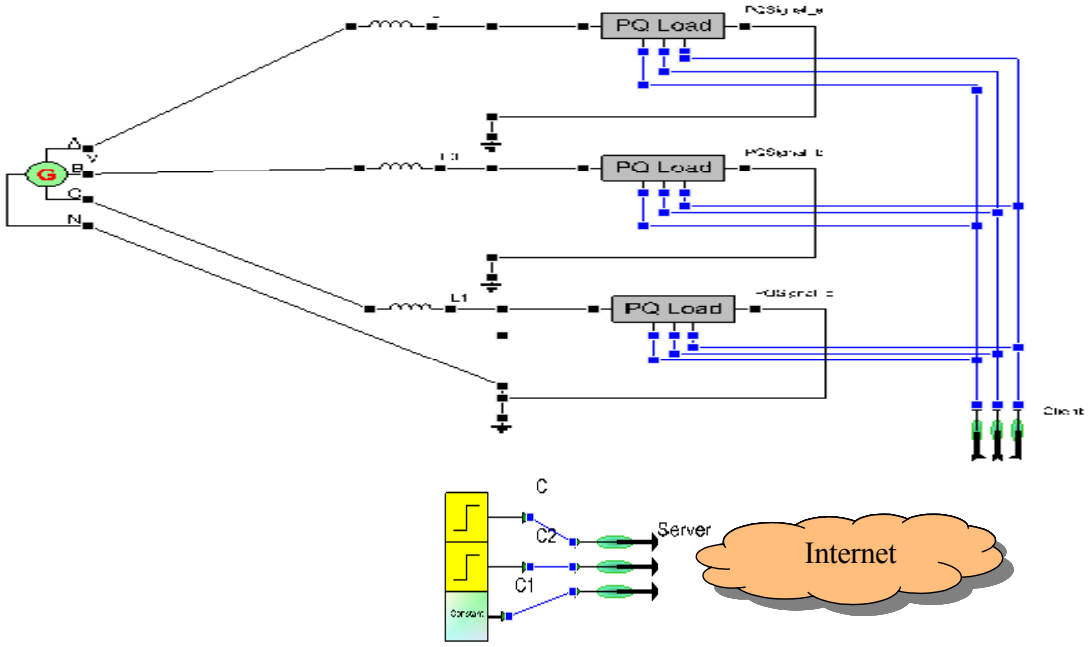


Figure 5.17 Signal Coupling Test in VTB with Changing Signal

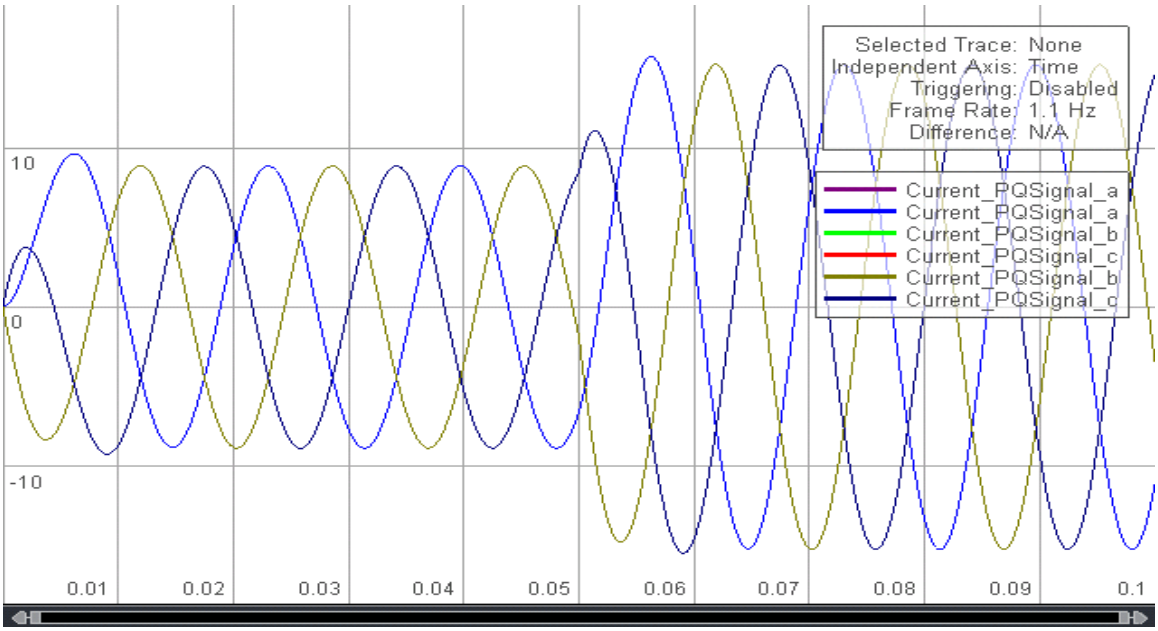


Figure 5.18 Signal Coupling Test Result in VTB with Changing Signal

For this simple system with changing signal source, the waveforms of the three-phase transmission line from distributed simulations match the waveforms from non-distributed simulation still exactly as observed from Figures 5.17.

5.4.3 Two-bus system with natural coupling and signal coupling

This section develops a test case is to test the agent models with both natural coupling and signal coupling in one simulation as shown in Figure 5.19. Figure 5.20 shows the simulation result of the transmission line current. The results of distributed test cases are compared with a single simulation to see the performance of distributed simulation.

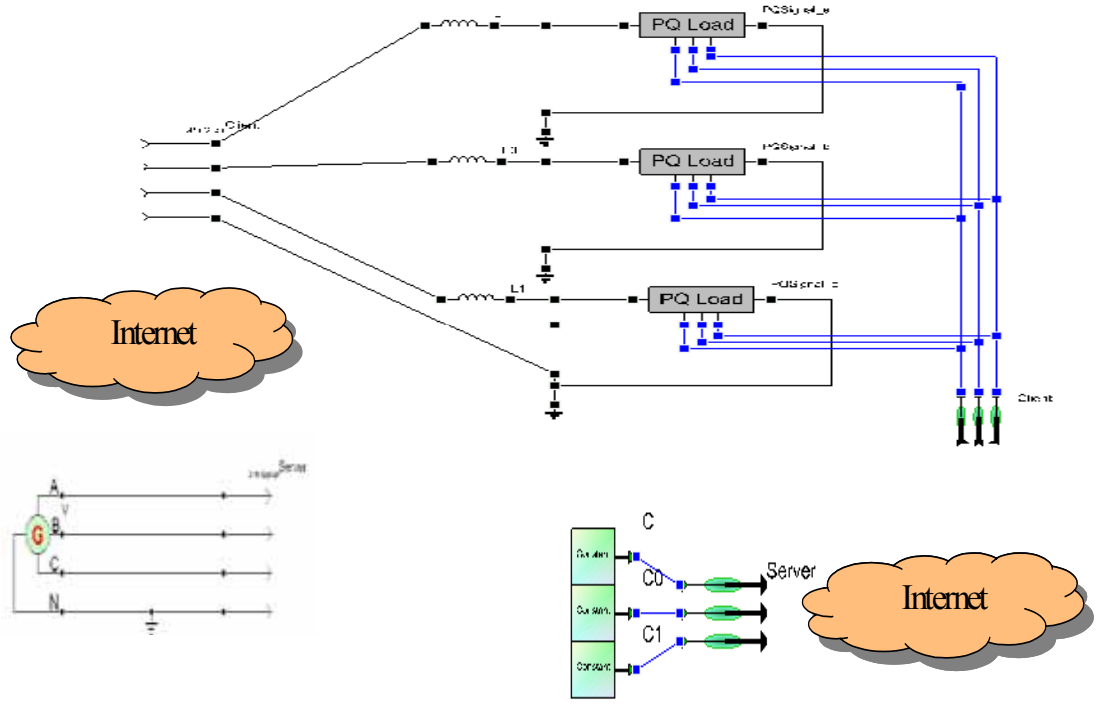


Figure 5.19 Natural/Signal Coupling Test in VTB

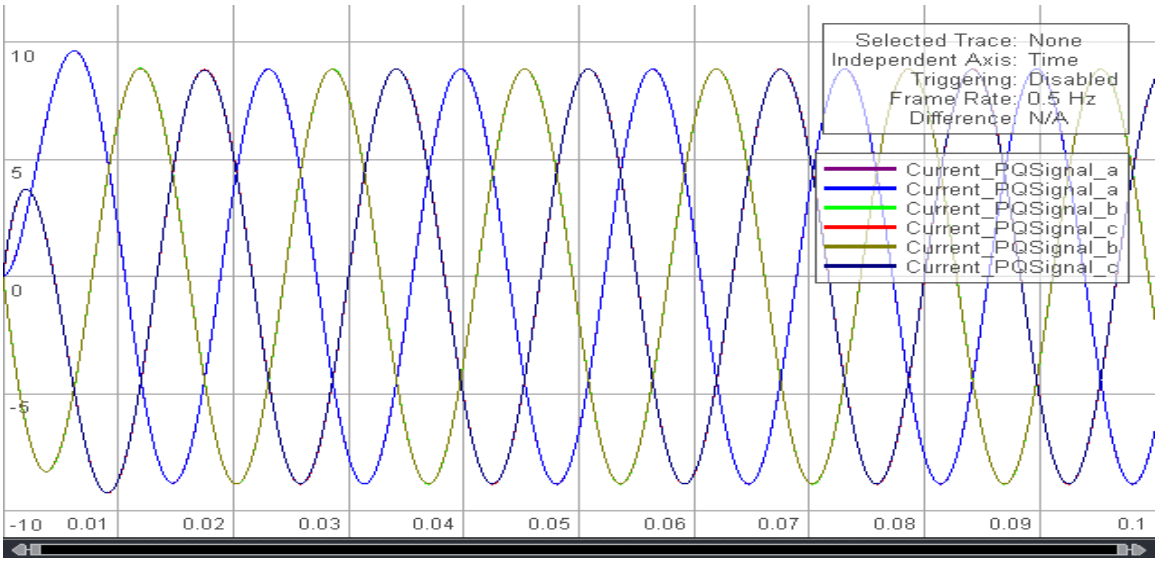


Figure 5.20 Natural/Signal Coupling Test Result in VTB

This simple system, for natural coupling, uses both linear coupling method and nonlinear in the test cases. The current of the three-phase transmission line, as shown in Figure 5.20, shows that the waveforms from distributed simulations match the waveforms from non-distributed simulation.

5.4.4 Steady state test with multiple source and multiple load

This section charts the development of a more complicated test case based on a new benchmark test system of a ship's distribution network described in paper [39]. This system is a 18-bus shipboard power system. It has six polynomial loads and four generators. This shipboard power system is configured in two zones. The two zones are weakly coupled through the transmission line 5 and 6. Therefore I selected this line as the decoupling point. Figure 5.21 shows the distributed simulation in VTB.

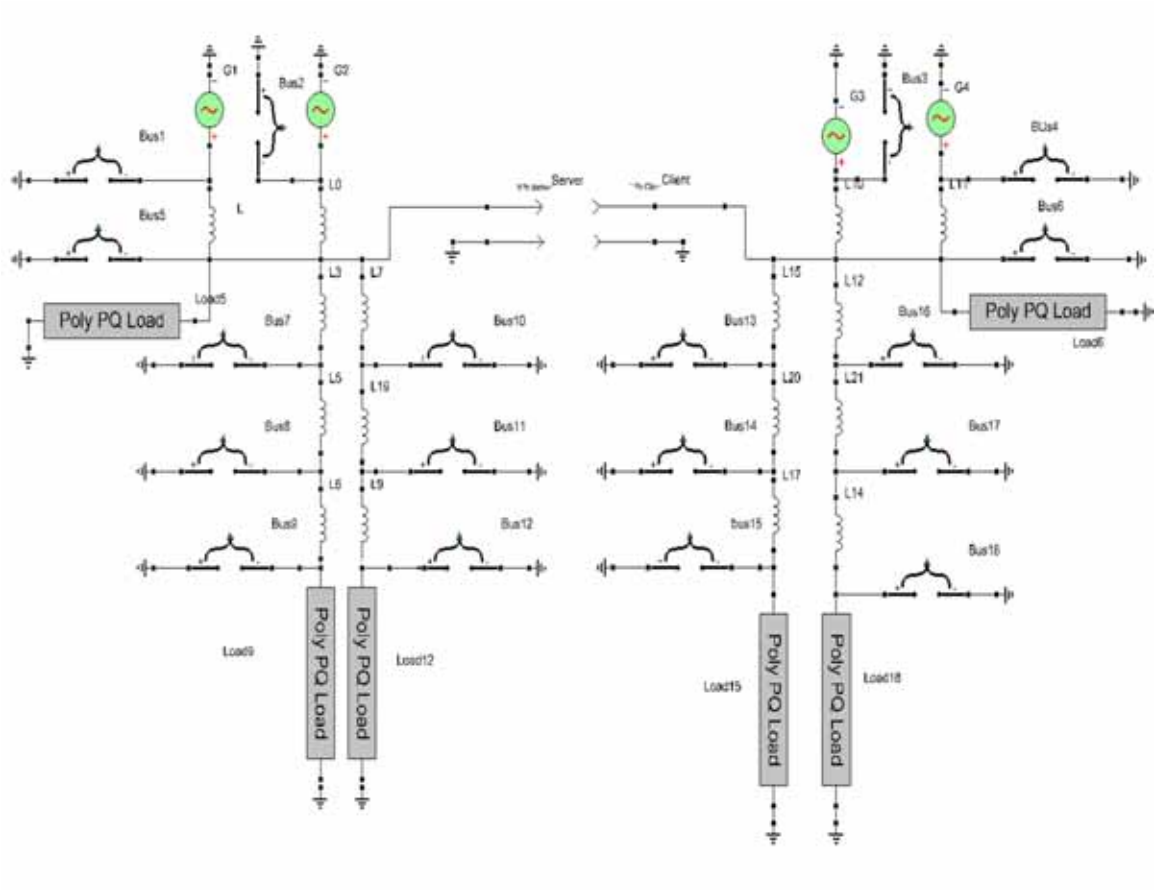


Figure 5.21 18-Bus Shipboard Power System Distributed Simulation

Here, the PQ load model is a non-linear model dependent on the terminal voltage's RMS value. This detailed model is described in previous chapter and in paper [40]; its performance has been validated. In this test, besides comparing the MAPE of current flowing through the tie line (from bus 5 to bus 6), the power flow solution is also compared between the distributed simulation and non-distributed simulation.

After processing the data, the MAPE is found within 0.5% deviation from the non-distributed simulation. Table 5.4 shows the power flow solution using VTB.

Table 5.4

Power Flow Solution from VTB

Bus No.	Voltage magnitude (pu)	Voltage angle (degree)	Real Power Generation, P (MW)	Reactive Power Generation, Q (MVar)
1	1.02	0	5.722	1.292
2	1.02	0	6.329	0.1424
3	1.02	0	6.112	0.1391
4	1.02	0	5.896	0.1335
5	1.0199	-0.0054		
6	1.0199	-0.0053		
7	1.0197	-0.0136		
8	0.9866	-10.5502		
9	0.9865	-10.5638		
10	1.0198	-0.0111		
11	0.9873	-10.6174		
12	0.9872	-10.6311		
13	1.0196	-0.0167		
14	0.9872	-10.4745		
15	0.9870	-10.4911		
16	1.0197	-0.015		
17	0.9851	-10.7343		
18	0.9849	-10.7502		

Compared to the result from the power flow solution in paper [39], the voltage magnitude is the same while the voltage angle deviation is within 0.014 degrees. This error diminishes, because in steady state, the error integrated to zero. The steady state

solution in distributed simulation matches the solution in non-distributed simulation [39]. The close voltage match indicates that the simulation confirms power flow analysis and can be used for transient analysis.

5.4.5 Results and discussion

Until now, different test cases are developed to test the agent models used for distributed simulation. Signal level coupling is straightforward. Its capability has been demonstrated with constant signal and changing signal. I achieved an extremely close match from distributed simulation and non-distributed simulation as expected.

Natural level coupling is more complicated. This chapter compares three methods, linear, non-adjusted stabilizing element and adjusted stabilizing element, to solve the system with the unknown external network. Numerical analysis proves the convergence of the algorithm. Test cases demonstrate the capability of natural level coupling agent model. Generally, a smaller time step makes the distributed simulation more accurate than non-distributed simulation. With a small enough time step, little difference occurs between non-adjusted stabilizing element and adjusted stabilizing element methods. But when the time step increases, an inappropriate stabilizing element will make the simulation diverge. Since there is limited information about the external system, guessing an appropriate stabilizing element is difficult. At this point, the advantage of adjusted stabilizing element is obvious. It can identify the optimal stabilizing element for the system to make the simulation converge. This characteristic is especially useful when the system is nonlinear or has a time varying element.

Also as noticed from the two-bus test cases for natural coupling model, a single load test has better performance than multiple loads. This difference is caused by the solver used by VTB, where voltage is taken as the state variable and is controlled within tolerance. The current of each element is computed separately. However, the truncation error can make the total current mismatch larger. The number of branches at the coupling point can affect the simulations results and produce a different MAPE.

The steady state test uses a test case of an electric ship. I achieves satisfactory results. The test demonstrates thenatural coupling model's capability to work with multiple sources and multiple loads. Testing the natural coupling model further with larger scale power system will be possible when VTB-Pro is released, as it can accommodate more nodes and is more stable.

5.5 Summary

This chapter an agent based distributed simulation to handle natural coupling and signal level coupling. It proposes a generalized VI coupling method dealing with the natural coupling. It analyzes the computation stability. I present a method of optimal adjustment of stabilizing element. I develop and demonstrate both the natural coupling and the signal level coupling, agent based model in Virtual Test Bed. I test the models with simple cases and present the test results. Agent models' performance in steady state and transient study is presented and analyzed to verify the correctness and applicability of the proposed algorithm.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

6.1 Conclusion

The ever-increasing need for computational power can be satisfied by the application of distributed simulation. In order to make the distributed simulation of a power system computationally efficient and reliable, this research focused on developing a distributed simulation algorithm for time domain simulation based on agent technology. This rresearch developed the traditional polynomial load model and RLC based PQ models to facilitate power system simulation in VTB, and proposed a mathematical model for the distributed algorithm. It analyzed the computation stability and presented a method of optimal adjustment of stabilizing element. For both the natural coupling and the signal level coupling, I developed and demonstrated agent-based models in VTB. are I presented and analyzed agent models' performances in steady state and transient study to verify the correctness and applicability of the proposed algorithm.

Test results of applying the proposed distributed simulation algorithm on transient and steady state power systems analysis have demonstrated that the algorithm and load models have the following salient features:

- The newly developed decoupled model will enable distributed simulation at both the natural coupling level and the signal coupling level, which can hide the model details from each side and has the potential to boost computation speed.
- The proposed distributed simulation method is computationally efficient and has guaranteed convergence. It can be easily implemented in power system simulation with linear and non linear elements.
- The newly developed load model enables the user to control the load remotely and monitor the test results over the network.
- The newly developed load model enables the user to perform transient analysis for a steady-state described element.

The main contribution of this dissertation is the development of a distributed simulation algorithm using agent technology that can perform system simulation without detailed internal network information, thus, only boundary measurements are needed. The proposed distributed method could be incorporated into parallel processing to speed up power system analysis. Also, the newly developed load model enables users to develop time domain simulation for traditionally steady state load models. The controllable load facilitates testing the control algorithm for load control and power system reconfiguration.

6.2 Future work

Although the proposed distributed simulation algorithm can perform time domain simulation at the single phase and three phase coupling point, the current method is used

for single point decoupling. An investigation of ways to extend the proposed VI coupling method to multiple coupling points will improve the flexibility of distributed simulation.

Furthermore, I recommend an investigation of applying the distributed simulation to the following areas is recommended:

- Incorporate the proposed distributed simulation into the design of parallel algorithm
- Develop a more intelligent agent, which is configurable with respect to types of coupling, (either natural or signal level), types of agent communication protocol, and is able to accept dynamic assignment of agent functionality.
- Extend the proposed algorithm to accommodate multiple system-decoupling schema.

REFERENCES

[1] K. Miu, V. Ajjarapu, K. Butler-Purry, D. Niebur, C. Nwankpa, N. Schulz and A. Stankovic, "Testing of shipboard Power Systems: A case for remote testing and measurement," *Proceedings of IEEE Electric Ship Technologies Symposium*, Philadelphia, PA, Jul. 2005.

[2] J. Wu and N. N. Schulz, "Experimental Design for Remote Hardware-In-the-Loop testing," *Proceedings of ASNE Reconfiguration and Survivability Symposium*, Jacksonville, Florida, Feb. 2005.

[3] S. Ayasun, S. Vallieu, R. Fischl, and T. Chmielewski, "Electric machinery diagnostic/testing system and power hardware-in-the-loop studies," *Proceedings of the 4th IEEE International Symposium on Diagnostics for Electric Machines*, Aug. 2003, pp. 361 – 366.

[4] S. Ayasun, S. Vallieu, R. Fischl, and T. Chmielewski, "Simulation-Stimulation Interface for Hardware in-the-Loop Studies", *MED POWER*, Athens, Greece, Nov. 2003.

[5] D. Potter, "Using Ethernet for industrial I/O and data acquisition," *Proceedings of the 16th IEEE Instrumentation and Measurement Technology Conference*, 1999, vol. 3, May. 1999, pp. 1492 – 1496.

[6] B. Teitelbaum, S. Hares, L. Dunn, R. Neilson, V. Narayan and F. Reichmeyer, "Internet2 QBone: building a testbed for differentiated services," *IEEE Network*, Vol. 13, No. 5, September – October 1999, pp. 8 – 16.

[7] A. Mohammed, E. Jones, H. Ogier, M. A. Vouk and Z. Dwekat, "DiffServ experiments: analysis of the premium service over the Alcatel-NCSU Internet2 testbed," *Proceedings of the 2nd European Conference on Universal Multiservice Networks*, April 2002, pp. 124 – 130.

[8] IEEE COMMIITEE REPORT, "Parallel processing in power systems computation," *IEEE Transactions On Power Systems*, Vol. 7, No. 2, pp. 629- 637, May 1992.

[9] K. W. Chant, A. R. Daniel, R. W. Dunnt and T. Berry, "A partitioning algorithm for parallel processing of large power systems network equations," *Proceedings of IEE 2nd International Conference on Advances in Power System Control, Operation and Management*, Hong Kong, Dec. 1993.

[10] D. E. Barry, C. Pottle and K. Wirgau, “A Technology Assessment Study of Near Term Computer Capabilities and Their Impact on Power Flow and Stability Simulation Program,” *EPRI-TPS-77-749 Final Report*, 1978.

[11] F. M. Brasch. J. E. Van Ness and S. C. Kang, "Evaluation of Multiprocessor Algorithms for Transient Stability problems,” *EPRI-EL-947*, 1978.

[12] F. M. Brasch, J. E. Van Ness and S. C. Kang, "Design of Multiprocessor Structures for Simulation of Power System Dynamics,” *EPRI-EL-1756*, 1981.

[13] A. M. Erisman, “Decomposition and Sparsity with Application to Distribution Computing,” Exploring Applications of Parallel Processing to Power System Analysis Problems, *EPRI-EL-566-SR*, 1977.

[14] J. Fong and C. Pottle, “Parallel Processing of Power System Analysis Problems via Simple Parallel Microcomputer Structures,” *EPRI-EL-566-SR*, 197.

[15] K. Lau, D. J. Tylavsky and A. Bose, “Coarse Grain Scheduling in Parallel Triangular Factorization and Solution of Power System Matrices,” *Proceedings of the IEEE PES Summer Meeting*, Minneapolis, July 1990.

[16] D. J. Tylavsky, “Quadrant Interlocking Factorization: A Form of Block L-U Factorization,” *Proceedings of IEEE PES Meeting*, 1986.

[17] A. Abur, “A Parallel Scheme for the Forward/Backward Substitutions in Solving Sparse Linear Equations,” *IEEE Transactions. on Power Systems*, Vol. 3, No. 4, pp. 1471 – 1478, 1988.

[18] F. L. Alvarado, D. C. Yu and R. Betancourt, “Partitioned Sparse A-l Methods [power systems],” IEEE Transactions on Power Systems, Vol. 5, No. 2, pp. 452 – 459, May 1990.

[19] W. F. Tinney, V. Brandwajn and S. M. Chan, “Sparse Vector Methods,” *IEEE Transactions on Power Apparatus and Systems*, Vol. 104, No. 2, pp. 295-301, Feb. 1985.

[20] J. E. Van Ness and G. Molina. “Multiple Factoring in the Parallel Solution of Algebraic Equations,” *EPRI EL-3893*, 1983.

[21] R. Betancourt and F. L. Alvarado, “Parallel Inversion of Sparse Matrices,” *IEEE Transactions on Power System,* Vol. 1, No. 1, pp. 74- 81, Feb. 1986.

[22] M. K. Enns, W. F. Tinney and FL. Alvarado, "Sparse Matrix Inverse Factors,” IEEE Transactions on Power Systems, Vol. 5, No. 2, pp. 466 – 473, May 1990.

[23] D. J. Tylavsky and B. Gopalakrishnan, "Precedence Relationship Performance of an Indirect Matrix Solver," *Proceedings of IEEE PES Summer Meeting*, Long Beach, CA, 1989.

[24] S. Y. Lee, H. D. Chiang, K. G. Lee and B. Y. Ku. "Parallel Power System Transient Stability Analysis on Hypercube Multiprocessors," *Proceedings of IEEE Power Industry Computer Applications Conference*, Seattle, May 1989.

[25] A. Gomez, and R. Betancourt, "Implementation of the Fast Decoupled Load Flow on a Vector Computer 8," *Proceedings of IEEE Power Engineer Society Winter Meeting*, Atlanta, 1990.

[26] F. L. Alvarado, "Parallel Solution of Transient Problems by Trapezoidal Integration," *IEEE Transactions on Power Apparatus and Systems,* Vol. 98, No. 3, pp. 1080-1090, May/Jun. 1979.

[27] V. B. Dmitriev-Zdorov, "Generalized coupling as a way to improve the convergence in relaxation-based solvers," *Proceedings of Design Automation Conference*, pp. 15 - 20, Sep. 1996

[28] V. B. Dmitriev-Zdorov, R. Dougal, V. Lyashev, M. Maksimov, V. Popov and E. Solodovnik, "Distributed Simulation of Electromechanical Systems in the Virtual Test Bed," *Proceedings of Power Electronics Specialists Conference*, Florida, Nov. 2001.

[29] S. Y. R. Hui and K. K. Fung, "Fast decoupled Simulation of large power electronic systems using new two-port companion link models," *IEEE Transactions on Power Electronics*, Vol.12, No.3, May 1997, pp. 462 - 473.

[30] G. Cokkinides and B. Beker, *VTB Model Developer's Guide*, Virtual Test Bed Project, Department of Electrical Engineering, University of South Carolina, Columbia, SC 29208, Available at http://vtb.engr.sc.edu/modellibrary/modeldev.asp, Mar. 2001.

[31] W. Gao, E. V. Solodovnik, and R. A. Dougal, "Symbolically Aided Model Development for an Induction Machine in Virtual Test Bed," *IEEE Transactions On Energy Conversion*, Vol. 19, No. 1, pp. 125 - 135, Mar. 2004.

[32] A. Deshmukh, F. Ponci, A. Monti, M. Riva and L. Cristaldi, "Multi-Agent System for Diagnostics, Monitoring and Control of Electric Systems," *Proceedings of International Conference on Intelligent Systems Application to Power Systems*, pp. 201-206, Nov. 2005.

[33] K. Huang, D.A. Cartes and S.K. Srivastava, "A Multi-Agent based Algorithm for Mesh-Structured Shipboard Power System Reconfiguration," *Proceedings of International Conference on Intelligent Systems Application to Power Systems*, pp. 188-194, Nov. 2005.

[34] Remote Procedure Call, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/rpc/rpc/generating_interface_uuids.asp.

[35] IEEE Task Force On Load Representation For Dynamic Performance, "Load representation for dynamic performance analysis," *IEEE Transactions On Power Systems*, Vol. 8, no. 2, May 1993, pp. 472- 482.

[36] R. A. Dougal, T. Lovett, A. Monti and E. Santi, "A Multilanguage environment for interactive simulation and development of controls for power electronics," *Proceedings of IEEE Power Electronics Specialists Conference*, Vancouver, Canada, June 17-22, 2001.

[37] Virtual Test Bed project website: http://vtb.ee.sc.edu/.

[38] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Computation of the Complex Fourier Series," *Mathematics of Computation*, Vol. 19, Apr. 1965, pp. 297-301.

[39] T. L. Baldwin and S. A. Lewis, "Distribution load flow methods for shipboard power systems," *IEEE Transactions on Industry Applications*, Vol. 40, No. 5, Sep. 2004, pp. 1183 - 1190.

[40] J. Wu, Y. Huang, W. Gao and N. N.Schulz, "Load Model in VTB," *Proceedings of IEEE Power Engineer Society General Meeting*, Montreal, Canada, Jun. 2006.