A Method for PMU-Based Reconfigurable Monitoring

Jerel Alan Culliss

Thesis submitted to the faculty of

Virginia Polytechnic Institute and State University

In partial fulfillment of the requirements for the degree of:

Master of Science

In

Electrical Engineering

Virgilio A. Centeno, Chair

Jaime De La Reelopez

Yilu Liu

October 27, 2009

Blacksburg, Virginia

Keywords: Phasor Measurements, PMU, State Estimation, Reconfigurable Monitoring

A Method for PMU-Based Reconfigurable Monitoring

Jerel Alan Culliss

Abstract

Given an increasing tendency towards distributed generation and alternative energy sources, the power grid must be more carefully monitored in order to ensure stability. Phasor Measurement Units (PMUs) provide very good observation of a small area of a network, but their relatively high cost prevents them from being deployed at every point. Therefore, to monitor an entire network, State Estimation is still required. By combining these two techniques, the accuracy and speed of power network monitoring can be improved. This thesis presents a method for achieving this goal from both hardware and computational perspectives. Practical considerations for PMU placement are discussed, such as instrument transformer calibration, and an algorithm is developed to apply this technique to any power system. The resulting method is termed reconfigurable monitoring - computationally isolated areas which may be grouped as necessary to allow for flexibility in power system monitoring.

Acknowledgements

I am forever grateful to my advisor and committee chair, Dr. Virgilio Centeno, for his guidance, inspiration, and diligence in keeping me on task. His comments and advice have greatly influenced not only my studies, but my life as a whole. I would also like to thank my other committee members, Dr. Jaime De La Reelopez and Dr. Yilu Liu. As an undergraduate, Dr. De La Ree's enthusiasm for teaching and seemingly endless knowledge of power engineering inspired me to continue with graduate studies. As a graduate student, Dr. Liu's passion for research and understanding of large-scale power system behavior helped me to diversify my knowledge and gain a more thorough understanding of my chosen field.

I would also like to thank all of my wonderful friends and family who have supported me throughout my education. Although there are too many people to list, each of them has had an impact on me for which I am extremely thankful. To my parents, I am eternally thankful for the opportunities you have granted me and all the support that you have given me along the way.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Background

In a power system, Wide Area Monitoring (WAM) is considered to be integral for power system stability [1]. Phasor Measurement Units, which allow for synchronized measurements, are considered to be one of the most important measuring devices in the future of power systems [2]. Historically, the technique of State Estimation has been crucial to the analysis of a power system. The output of a state estimator allows such functions as contingency analysis, load forecasting, and constrained optimal dispatch to be performed in real time.

1.1.1 Phasor Measurement Units

Phasor Measurement Units, or PMUs, are power system devices which are able to measure voltage and current, and then calculate relative phase angle differences. The general term for this type of measurement is *synchrophasor*. Algorithms for the computation of a phasor have existed since the 1970's, when Symmetrical Component Distance Relays (SCDRs) were developed. While this directly provides a great deal of information about the status of a particular bus on the power grid, it is the Global Positioning System (GPS) which makes the modern PMU truly powerful. In the mid 1980's, the development of GPS allowed researchers at Virginia Tech to synchronize phasor measurements using time pulses with a precision of one microsecond (μs). This allowed for accurate and meaningful wide area measurements over large portions of a system. Following this, PMUs became a commercial product for all major IED manufacturers in the power industry. To ensure that PMUs from different manufacturers have a high degree of operability, the IEEE SYNCHROPHASOR [3] standard, IEEE 1344-1995 was developed. The unprecedented increase in communication technology in the decade following 1995 led the IEEE to revise and update the synchrophasor standard, creating c37.118-2005 [4]. PMUs are now

considered to be a critical and important measurement device with many potential applications for wide area stability[2][5]. Unfortunately, the cost of these devices, and their installation, is currently quite high, which acts as a practical limit on the number that may be deployed in a given system. Additionally, large-scale PMU deployment requires that the grid also has a sufficiently developed communication infrastructure. As a result, current synchrophasor research, such as the concept of reconfigurable monitoring presented here, is focused on maximizing the benefit of PMUs while keeping hardware costs to a minimum.

1.1.2 State Estimation

State estimation is a method of using available data to produce the best possible estimate of the system state. Measurements from the power system are used to determine the state variables, which are voltage magnitude and angle (V and θ) in the case of a power system. Due to the way in which state variables are calculated, state estimation greatly enhances the monitoring capability of a power system in two ways. First, not every point in a system must be measured directly; measurements on nearby buses often provide necessary information. Second, by the same principle, neighboring measurements provide a level of redundancy and help ensure accuracy. For these reasons, State Estimators are now a central part of all Energy Management Systems (EMS) [6].

Most state estimators essentially utilize a method of solving a system of non-linear, simultaneous equations as they are applicable to the power system. On the highest level, this is simply a Newton-Raphson method, although robust state estimation packages include statistical analysis and even Lagrangian relaxation techniques. The state variables of the system are contained in a vector and consist of the voltage magnitude and voltage angle for every bus, typically represented in Per Unit (P.U.) The easiest way to perform this computation is with

matrices, which are used as data storage and as computational tools. The key matrices in this computational process are the Jacobian matrix (H), which relates measurements to their state variables, and the gain matrix (G), which ensures that the P.U. order of magnitude is maintained. Another advantage of using matrices for state estimation is that they are massively parallel, and supporting multiple cores or even multiple processors is a possibility. A traditional state estimation algorithm is presented in more detail in Chapter 2.

1.2 Motivation and Objective

Phasor Measurement Units are excellent devices for monitoring sections of a power system individually, but the high cost associated with them is a strong limiting factor in their deployment. As mentioned previously, state estimation is currently used to solve the steady-state power system variables (V, θ) when measurements at every bus are not feasible. Although a state estimator with sufficient computational power and supporting communication infrastructure may be computed in seconds, this is not always the case. Potential communication delays and measurement uncertainty in a power system may limit the computation frequency of a full system solution to every few minutes. In a market where the desire for renewable energy and distributed generation is steadily increasing, this time frame may not be acceptable. It has been noted that upgrades to the monitoring and control tools available would greatly enhance the situational awareness of system operators to respond to contingencies[7]. Additionally, more frequent measuring capability may allow for improved economic operation of the power system. Given that PMUs are powerful measurement tools and are being installed in many power systems, computational methods which further enhance their functionality can have large benefits on modern grids.

The objective of this thesis is to introduce and explain the technique of reconfigurable monitoring as applied to a power system. The goal of reconfigurable monitoring is to provide a computational method to system operators which allows for flexible, resilient, and asynchronous state estimation capability. This is not intended to replace traditional state estimation, but rather to supplement it - this method helps to fill observability gaps which exist between full system state estimation executions. Additionally, contingencies may occur where full system state estimation is not feasible - such as a black start condition - but reconfigurable monitoring will allow for some degree of observability. To do this, PMU data must be successfully integrated into a state estimator, and the accuracy of the measurements and quality of the data must be ensured. The system model must also be computationally isolated in order to allow for a divided, asynchronous state estimation process. Therefore, the following topics are discussed in this thesis.

- PMU functionality as applied to a state estimator and use of synchrophasors to allow for computational isolation of a system model.
- Calibration of instrument transformers and bad data detection methods to ensure state estimation measurement accuracy.
- Installation considerations for PMUs, including an algorithm to optimally place PMUs in a power system for the purposes of reconfigurable monitoring as well as general observability.

1.3 Thesis Organization

This thesis is comprised of six chapters. Chapter 1 provides background on PMUs and state estimation, and presents the motivation and objective for reconfigurable monitoring. Chapter 2 elaborates on the calculations involved in state estimation, and explains how proper

handling of PMU data can serve to reduce the complexity of the matrices. The concept of computational isolation/islanding is explained. Chapter 3 focuses on practical considerations which must be addressed for reconfigurable monitoring to work. These include proper calibration of instrument transformers and state estimation considerations such as bad or unreliable data. Chapter 4 presents a greedy algorithm designed to optimize PMU placement for the purpose of reconfigurable monitoring. The system model and greedy algorithm are discussed, and an example of the algorithm's system traversal is given. Chapter 5 consists of three different applications of reconfigurable monitoring: system observability, state estimation execution time reduction, and dynamic state estimation fault tolerance. The IEEE 118 bus test system, 300 bus test system and the Brazilian ONS transmission system are used in these tests. Chapter 6 summarizes the work presented in this paper and makes recommendations for future research.

Chapter 2. Concepts of Reconfigurable Monitoring

2.1 Introduction

At a high level, reconfigurable monitoring requires that a system can be separated into multiple coherent parts, although these divisions are purely computational, and need not correspond to electrical islanding. To explain how a system can be computationally divided into multiple observable parts, a deeper look into the state estimation process is required. Of key consideration here is how PMUs are utilized - there is a large difference when using PMUs as simple measurement devices as opposed to treating them as computational borders. The discussion which follows will emphasize the importance of the PMU in the latter role and how this can be used to reduce the computational complexity of the state estimation process. This allows for a state estimator to be executed more frequently and in a parallel or asynchronous manner, providing system operators with wide-area observability when traditional state estimation is not possible.

2.2 Traditional State Estimation

A state estimator requires three types basic types of data: system topology, measurements, and the state variables. Strictly speaking, the list of state variables can be determined from the system topology. As previously mentioned, these state variables are the voltage magnitude and angle at every bus in the system, with a provision made for a reference bus. Therefore, the state variable vector $[E]$ contains $2n_b - 1$ state variables, where n_b is the number of buses in the power system. The measurement vector $[z]$ consists of quantities such as real and reactive power flows and injections, bus voltage magnitude, and line current magnitude. A pre-processing step removes bad data before it is included in the measurement vector. In addition to the scanned measurement vector $[z]$, a vector $[\varepsilon]$ containing the error or uncertainty

of each measurement must be known. From this error vector, the covariance matrix $[W]$ can be created. The relationships between these fundamental quantities are shown below:

$$[z] = [h(E)] + [\varepsilon] \tag{2.1}$$

$$[W] = diag(\varepsilon^{-1}) \tag{2.2}$$

where $h(x)$ is the set of non-linear functions of the state vector $[E]$.

Using these relationships, the goal of the iterative process is to start with an initial guess and then converge on a system estimate. As mentioned above, the function $h(x)$ is the set of non-linear functions which relates the measurements to the state vector. The functions themselves are provided in Appendix A. Therefore, the initial step of each iteration is to determine the difference between the measured values and the estimated ones. This difference is given by:

$$[z - h(E^k)] \tag{2.3}$$

where $[E^k]$ is the current estimate (k^{th} iteration) of the state variables.

After the difference between the measurement and estimate are calculated, the Jacobian matrix$[H]$ is calculated from the partial derivatives of the non-linear measurement functions with regard to the state variables. Explicit functions are provided in Appendix A; the Jacobian is essentially the following formulation:

$$[H(E)] = \left[\frac{\partial h(E)}{\partial (E)}\right] \tag{2.4}$$

Each iteration of the state estimator calculates the following iterative change to the state variable vector and then adds it to the most recent estimate.

$$\Delta E^{k+1} = H^T E^k W[z - h(E^k)]G(E^k)^{-1} \tag{2.5}$$

where the gain matrix, $[G(E^k)]$ is given by:

$$[G(E^k)] = [H^T(E^k)W^{-1}H(E^k)]^{-1} \tag{2.6}$$

It should be noted that $[G(E^k)]$ is square matrix which consists of triangles such that $[G(E^k)] = LL^T$. Computationally, this means that a Cholesky decomposition may be used to invert the matrix, providing both a speed advantage as well as numerical stability. The estimator is considered to have converged when the iterative change in an objective function $J(x)$ has reached a target low value. The calculation for this objective function is as shown:

$$J(x) = [z - h(x)]^T W[z - h(x)] \tag{2.7}$$

where the state vectors $[E^k]$ and $[E^{k-1}]$ are used in place of x to determine the convergence of the estimator. After convergence, the state variables should represent the system for the timestamp in which the estimation was performed. This is why state estimation is generally accepted as a steady-state operation - it does not react quickly enough to changes in the system to allow operators to react to extremely short disturbances[6].

2.2.1 PMUs in State Estimation

As explained in the previous chapter, PMUs allow for direct synchronized measurements of magnitudes and angles for both phase voltages and line currents. The voltage measurements (V, θ) directly correspond to two variables of the state estimator. The combination of voltage and current measurements taken by the PMU also provides real and reactive power flows, and as a result, the net real and reactive power injection at the bus. Figure 2.1 illustrates the power system measurements that a PMU can provide.

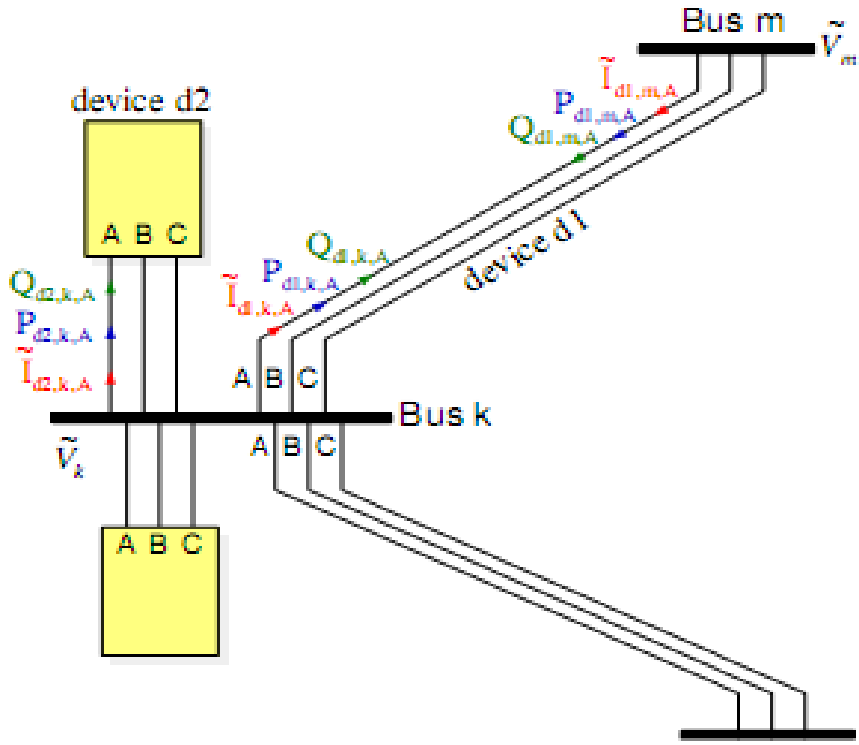

Figure 2.1: PMU and Associated Measurements

The implication of these measurements means that an installed PMU effectively provides measurements of the state variables for a group of buses. A PMU will *directly* measure the state variables of the bus that it is installed on, and *indirectly* measure the state at buses which are directly connected to it. This does however, assume that enough metering or relaying transformers are available and that the PMU is capable of supporting the amount of branch connections. If this is possible, then a solution may be devised where the entire power system is observable entirely through PMU measurements, hardware permitting. Many algorithms have been created to optimize PMU placement for this purpose, although the scale of large power systems means that no single "optimal" solution has been developed.

2.2.2 Decoupled State Estimation

Given that the size of the matrices used in state estimation increase exponentially with system size, various computational and mathematical shortcuts are used in an effort to reduce the computation time. Common shortcuts include sparse matrix techniques and computational shortcuts such as the Cholesky decomposition for matrix inversion. A PQ-decoupled formulation of the state estimator may also be used to speed up the computation. The formulation separates the measurements into two parts[8]:

Real power (P) related measurements, such as real power bus injections and real power line flows. If bus voltage angle data is available, then it is also included in this group.

Reactive power related (Q) measurements, such as reactive power bus injections, reactive power line flows, and bus voltage magnitude measurements.

The justification for this partition is due to the fact that real power flows are more sensitive to changes in voltage angle, whereas reactive power flows are more sensitive to changes in voltage magnitude. This approximation is valid for most high voltage transmission systems, as branch susceptance is much higher than branch conductance and any change in voltage angle on a given branch will be small. Mathematically, $B_{ij} >> G_{ij}$ and $\theta_{ij} \approx 0$ where i and j are two adjacent buses, B is the branch susceptance, G is the branch conductance, and θ is the change in voltage angle.

The result of the decoupled formulation is that the number of utilized cells within the matrices is reduced by 50%. Since such a state estimator is designed to handle the sparse regions within the matrices, the computation time for the algorithm is likewise approximately reduced by 50%.

Table 2.1 Standard and Decoupled Matrix Formulations

Standard	PQ-Decoupled
$W = diag(\varepsilon^{-1})$	$W_P = diag({\varepsilon_P}^{-1})$ $W_Q = diag({\varepsilon_Q}^{-1})$ $W = \begin{bmatrix} W_P & 0 \\ 0 & W_Q \end{bmatrix}$
$H = \begin{bmatrix} \frac{\partial P}{\partial \theta} & \frac{\partial P}{\partial V} \\ \frac{\partial Q}{\partial \theta} & \frac{\partial Q}{\partial V} \end{bmatrix}$	$H = \begin{bmatrix} \frac{\partial P}{\partial \theta} & 0 \\ 0 & \frac{\partial Q}{\partial V} \end{bmatrix}$
$G = [H^T W H]$	$G = \begin{bmatrix} \frac{\partial P}{\partial \theta}^T W_P^{-1} \frac{\partial P}{\partial \theta} & 0 \\ 0 & \frac{\partial Q}{\partial V}^T W_Q^{-1} \frac{\partial Q}{\partial V} \end{bmatrix}$

While PQ-decoupling offers a significant speed increase, there are two caveats which should be noted. First, there may be conditions where the network parameters or operating conditions violate the mathematical assumptions which have been made. While uncommon in high voltage transmission systems, such a situation could result in an inaccurate solution. Second, branch current magnitude measurements are not included in either decoupling group - these measurements contain both a sensitive real and reactive component. In systems where branch current magnitude measurements are required for full observability, PQ-decoupled state estimation may not be used[6].

On the surface, this would seem to limit the ability of a PMU to provide useful measurements, as many of the measured values are current magnitudes. However, since the phase angle is also being measured, a pre-processing step may be used to calculate flow measurements from the synchrophasor data. In doing so, care must be taken to ensure that the

combined error is properly calculated. The overall result is that with some amount of pre-processing, significant speed increases can be achieved by reducing the complexity of the matrices involved.

2.3 Computational Isolation/Islanding

Regardless of how efficient a state estimator may be, it is limited to solving all of its equations at the same time, in essence it cannot partially solve the power system. This means that there is an upper limit on execution frequency due to both communication delays as well as the runtime of the estimator itself, which increases at a non-linear rate as the system increases in size. It should be noted that utility companies only perform state estimation on their own portion of the system, which makes sense as utilities do not necessarily share topology or measurement data. To establish a computational border such as this, a system equivalent is used. A typical equivalence consists of three subsystems: the "internal system" to be studied, the "external system" which is everything else, and then the buses which provide connectivity between the subsystems. A diagram illustrating this subsystem assignment is shown below.

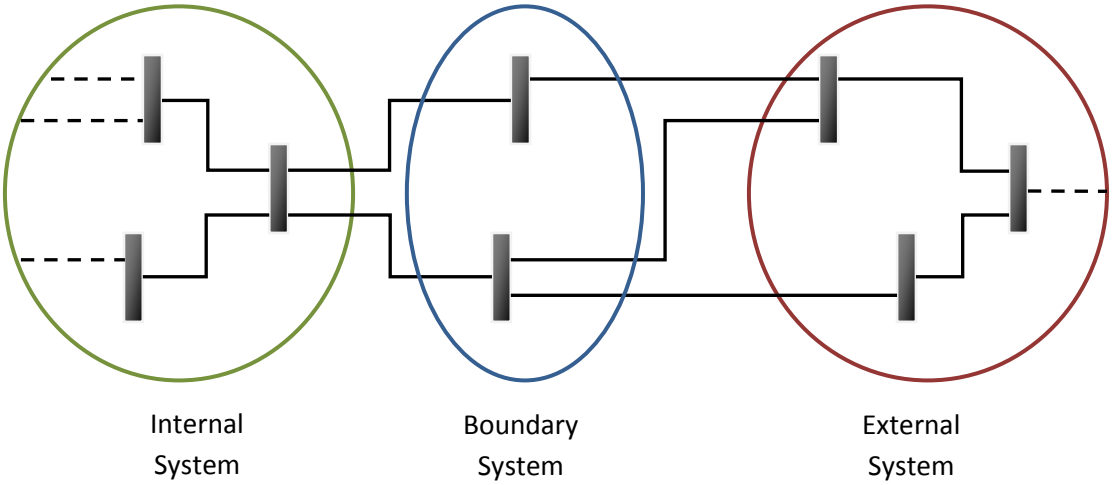

Figure 2.2: Bus Groupings for System Equivalencing

When groups of buses are defined as they are here, then the traditional measurement equation, $[z] = [h(E)] + [\varepsilon]$ can instead be refined into a more precise form:

$$\begin{bmatrix} z_i \\ z_b \\ z_e \end{bmatrix} = \begin{bmatrix} h_{ii} & h_{ib} & 0 \\ h_{bi} & h_{bb} & h_{be} \\ 0 & h_{eb} & h_{ee} \end{bmatrix} \begin{bmatrix} E_i \\ E_b \\ E_e \end{bmatrix} + \begin{bmatrix} \varepsilon_i \\ \varepsilon_b \\ \varepsilon_e \end{bmatrix} \tag{2.8}$$

As the only concern here is the internal system, only the top two rows of this equation are of importance. However, the middle row shows mathematically what is apparent in the system diagram - the internal and external subsystems are not completely isolated. The measurement equation for the boundary subsystem reduces to:

$$z_b = h_{bi}E_i + h_{bb}E_b + h_{be}E_e + \varepsilon_b = z_{bi} + z_{bb} + z_{be} + \varepsilon_b \tag{2.9}$$

An assumption is made that the term $z_{be} = h_{be}E_e$ can be rewritten as $z_{be} = \hat{h}E_b$ where $\hat{h} = \frac{z_{be}}{E_b}$. The component z_{be} may be estimated by subtracting all other components from the complete measurement vector, z_b. Using this estimation, and substituting back into Equation 2.8 (while ignoring the external system), the resulting measurement function is:

$$\begin{bmatrix} z_i \\ z_b \end{bmatrix} = \begin{bmatrix} h_{ii} & h_{ib} \\ h_{bi} & (h_{bb} + \hat{h}) \end{bmatrix} \begin{bmatrix} E_i \\ E_b \end{bmatrix} + \begin{bmatrix} \varepsilon_i \\ \varepsilon_b \end{bmatrix} \tag{2.10}$$

Alternatively, it is possible to have a load-generation equivalent bus connected to all border buses on the topology level. This is essentially what the matrix reduction is doing - the equations given by h_{be} are essentially the flows between the boundary and external systems[9]. While this method allows for a system to be divided into pieces, the computation quickly increases in complexity as the number of subsystems increases. Dividing a system into more than 3 subsystems in this manner requires much more up-front computation, especially when the

subsystems have more than a single border. While this may allow for pre-planned contingencies, such as electrical islanding, to be handled, it is pre-configured monitoring rather than reconfigurable. The difference in terminology is that *reconfigurable* implies reactive online capability as well as pre-configured contingencies.

2.3.1 PMUs as Borders

As shown in the system equivalencing example in the previous section, the key quantities involved in computational isolation are the power flows entering and leaving the target subsystem. A PMU has the ability to measure both the magnitude and angle of bus voltages as well as line currents, meaning that it can effectively measure all flows relating to the bus on which it is installed. The timing pulse from the GPS keeps accuracy within 250 nanoseconds, and allows for local timing pulses to be synchronized at better than 1 microsecond precision. In a 60 Hz system, this translates into a 0.02 degree error at most[10]. This means any measurement error introduced by the PMU itself is very small, especially compared with any other measurement device in the system. By comparison, nearly the entire error associated with synchrophasor measurements instead results from the instrument transformers which are connected to the PMU. If these devices are properly calibrated, or their systematic error is known, then the synchrophasor measurements provide the solution for the state variables of a border bus as well as accurate real and reactive power flows to the subsystems on either side. Instrument transformer error and calibration is discussed in detail in Chapter 3. For now, the following examples assume that the instrument transformers connected to the installed PMUs are properly calibrated or their calibration error is otherwise known.

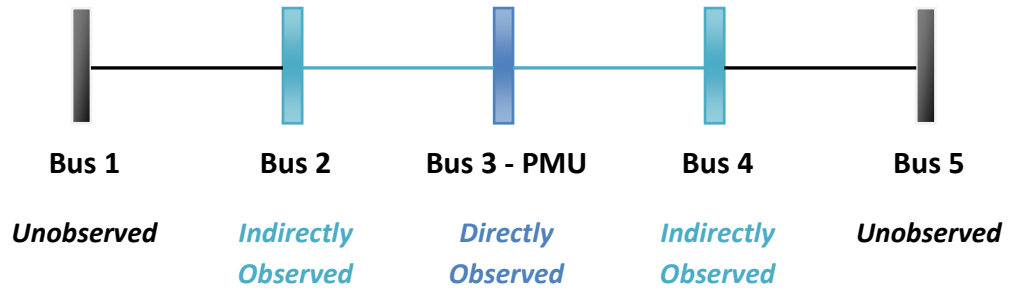

Figure 2.3: Phasor Measurement Unit Observability

The diagram above illustrates the observability granted by a PMU. Using this, there are two different ways in which a PMU can act as a computational border. The first case only considers the bus that a PMU is installed on to act as a border. In this case, bus 3 is removed and the result is the following system:

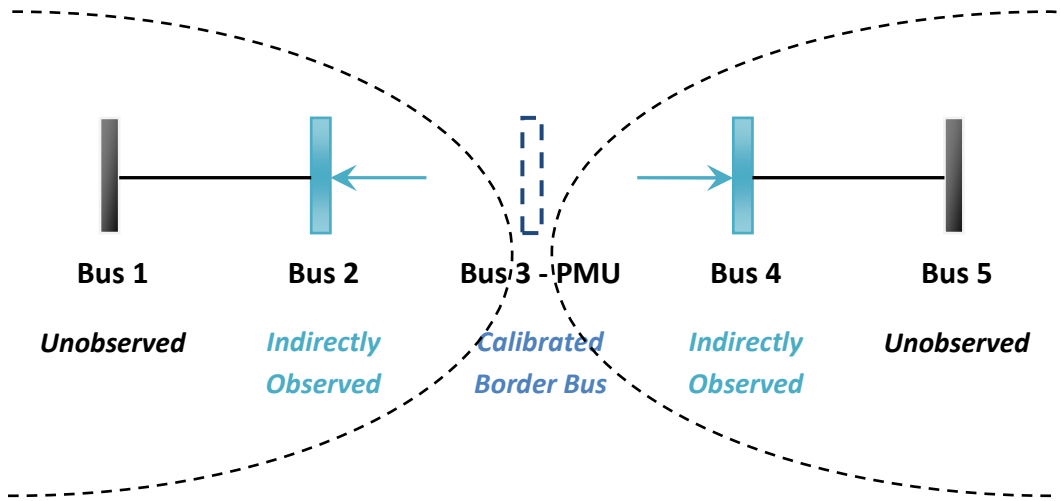

Figure 2.4: Phasor Measurement Unit as a Single-Bus Border

The resulting subsystems can be analyzed individually. The arrows on buses 2 and 4 in the above diagram indicate implied injection measurements, and can be used if real or reactive power flow measurements were present in the original system. This situation also applies if the error is known on the voltage transformers but unknown on the current transformers, making the indirect synchrophasor measurements too unreliable for the purposes of computational isolation.

The second case considers that the current transformers are also either properly calibrated or their errors are known. If this is achieved, then the PMU also indirectly observes the state variables of all adjacent buses, or at least those connected by branches with synchrophasor current measurements. This is an important distinction, as hardware considerations may not allow for all line currents to be measured at a given bus, or as presented in the single-bus case, the current transformers cannot be calibrated. Assuming that the line currents are measured, and that the errors on the instrument transformers are known, the border expands to encompass the group of buses adjacent to the PMU:

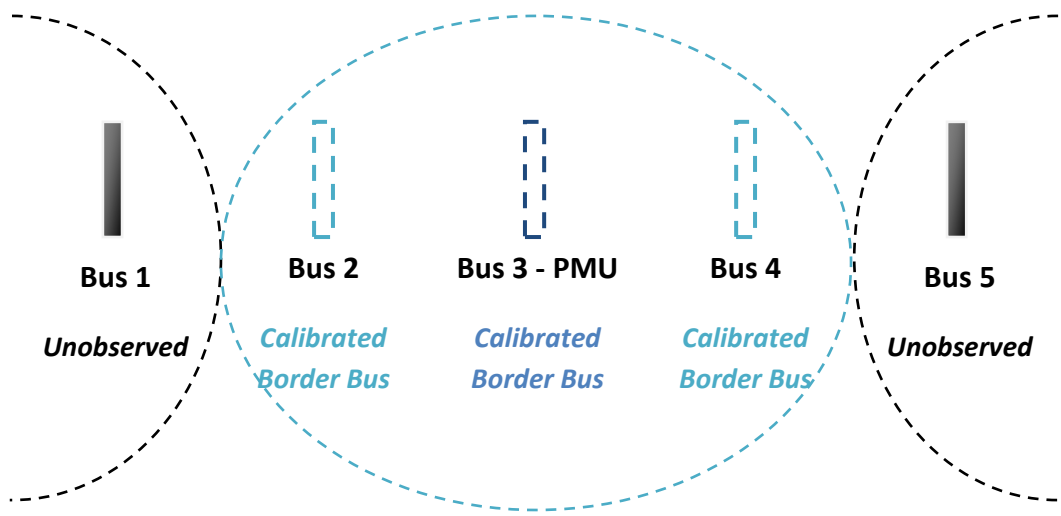

Figure 2.5: Phasor Measurement Unit as a Multi-Bus Border

The principle is the same in either case - if the synchrophasor data can be trusted, then the state variables are directly known and estimation of the PMU-observed buses is not required. In either the single or multi-bus border case, the set of measurement equations is:

$$\begin{bmatrix} z_i \\ z_b \\ z_e \end{bmatrix} = \begin{bmatrix} h_{ii} & h_{ib} & 0 \\ 0 & 1 & 0 \\ 0 & h_{eb} & h_{ee} \end{bmatrix} \begin{bmatrix} E_i \\ E_b \\ E_e \end{bmatrix} + \begin{bmatrix} \varepsilon_i \\ 0 \\ \varepsilon_e \end{bmatrix} \tag{2.11}$$

The assumption is made that the quantization and timing error directly associated with the PMU may be ignored. Since these errors are much smaller than any other error in the system, including the instrument transformers[11], this approximation is not unfounded. The similarity to Equation 2.8 is apparent, with the notable difference that the boundary measurements are not estimated; $z_b = E_b$. This allows for a direct simplification of the equations as follows:

$$z_i = h_{ii}E_i + h_{ib}E_b + \varepsilon_i \tag{2.12}$$

$$z_e = h_{ee}E_e + h_{eb}E_b + \varepsilon_e \tag{2.13}$$

Compared with the system equivalencing method discussed previously, this has the advantage of further reducing the number of equations while also easily accommodating for a division of a system into more than two subsystems and a single boundary. As every PMU-based border is completely isolated, measurements from the adjacent subsystems are not required. This allows for the subsystems to not only be solved independently, but asynchronously as well.

2.3.2 State Estimation Complexity with Computational Islands

With computational islands established for a power system, and the measurement equations simplified as shown in the previous section, the corresponding matrices involved in state estimation are likewise simplified. To illustrate this, the IEEE 14 bus test system is presented and the complexity of the system's state estimation matrices is discussed. The algorithm used for determining placement of the PMUs in this system is explained in chapter 4. The system depicted uses only single-bus borders - the 14 bus system is too small to make use of multi-bus borders, and their inclusion would result in a system which is nearly fully observed through synchrophasor measurements.

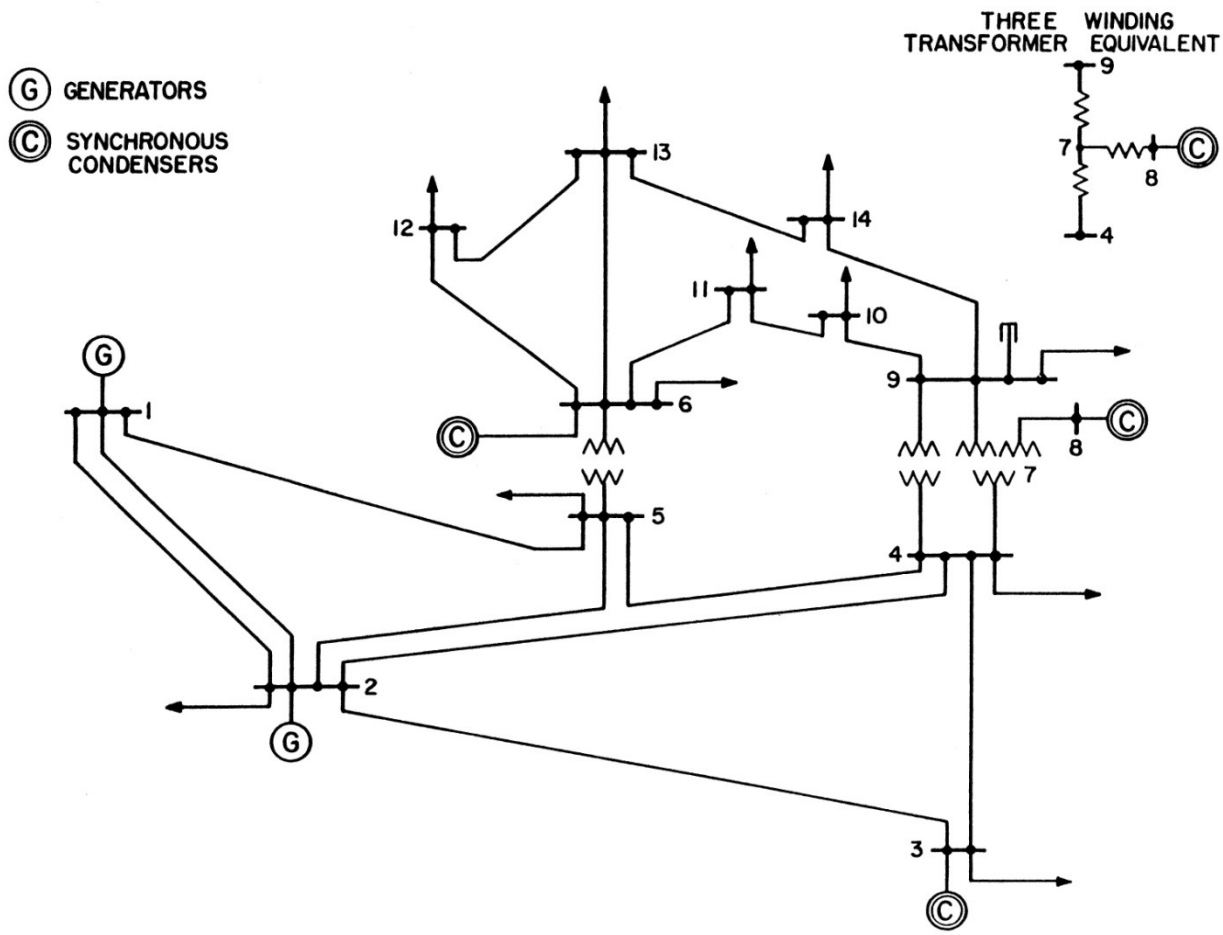

Figure 2.6: IEEE 14 Bus Test System

The system as shown above has 27 state variables - 13 voltage angles and 14 voltage magnitudes. For simplicity, it is assumed that Bus 1 serves as the reference angle and thus $\theta_1 = 0$. For measurements, an *injection only* case is used - the set of measurement equations consists of 14 voltage magnitudes and 14 power measurements. The sizes of the Jacobian (H) and Gain (G) matrices serve as a good indicator of overall computational complexity[6]. Despite the typical sparseness of the Jacobian matrix, the gain matrix G is dense and derived from H, and so sparse matrix techniques cannot be used when calculating the gain. The size of each matrix is calculated as follows:

$$H_{size} = N_{measurements} \times N_{state\ variables} \tag{2.14}$$

$$G_{size} = N_{state\ variables} \times N_{state\ variables} \tag{2.15}$$

With no PMUs in the system, the calculations are $H_{size} = 28 \times 27 = 756$ cells and $G_{size} = 27 \times 27 = 729$ cells. If a PQ-decoupled form is used instead, then there are two separate Jacobian matrices, and each one is $H_{size} = 14 \times 13 = 182$ cells, meaning that there are 364 cells in total. The two gain matrices are respectively reduced to 169 and 196 cells, 365 in total .

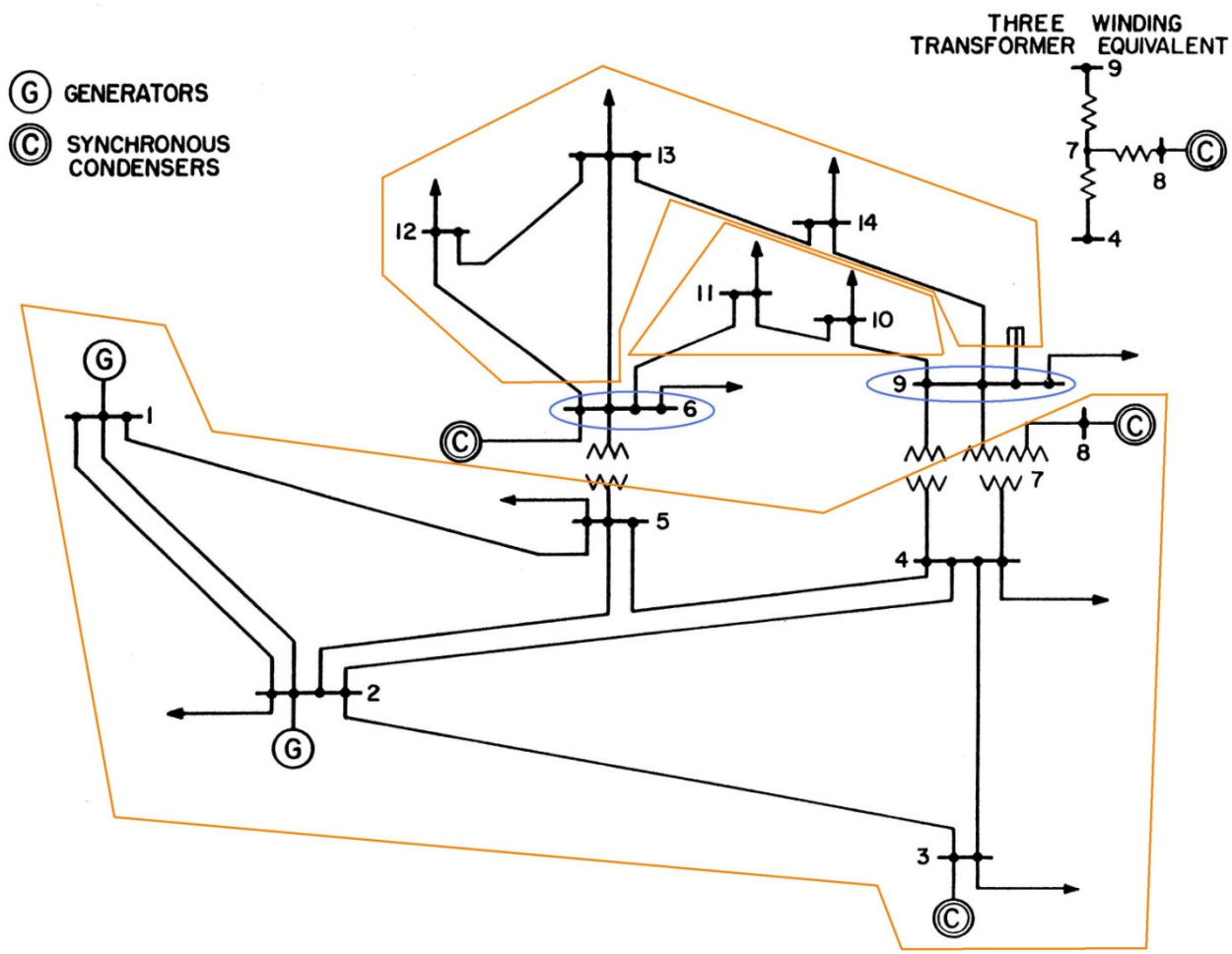

Figure 2.7: IEEE 14 Bus Test System with PMUs as Computational Borders

With PMUs placed on buses 6 and 9, three computationally isolated areas, termed computational islands, are created. Area 1 consists of buses 1 through 8, Area 2 consists of buses 10 and 11, and Area 3 consists of buses 12, 13, and 14. The buses with installed PMUs are not included in any of the subsystems. As in the example with no PMUs, the *injection-only* case is considered here, and the reference bus is still Bus 1. With three subsystems to consider, the complexity calculations are:

$H_{size1} + H_{size2} + H_{size3} = 16 \times 15 + 4 \times 4 + 6 \times 6 = 292$ cells in total for all three Jacobian matrices and $G_{size1} + G_{size2} + G_{size3} = 15 \times 15 + 4 \times 4 + 6 \times 6 = 277$ cells for all three Gain matrices.

Since this is an injection-only case, PQ decomposition is possible, reducing the total number of cells by 50% - 146 for the total of all 6 Jacobian matrices. The total for the gain matrices is reduced to 139 cells. At this point, the largest matrix is an 8x7 Jacobian, and it has high density - very few of the cells are empty. Due to the computational overhead required in setting up the measurements, and executing the cycles, PQ decomposition may not be beneficial in situations such as this.

The purpose of this example is to illustrate the relative reduction in computational complexity due to computational islanding - a real system of this size would not truly benefit from such a method. The system is too small to warrant it, and the computational overhead would negate any potential speed increase gained by matrix reduction. The key is that as a system increases in size, the effect of dividing a system with computational islanding is worth more - larger systems allow for more computational islands.

Chapter 3. Practical Considerations

3.1 Introduction

Chapter two introduced the basic concepts behind reconfigurable monitoring, including the use of a PMU as a computational border. In doing so, a key assumption was made, that the instrument transformers associated with the PMUs could either be calibrated or have their errors known. This chapter will discuss instrument transformers, their associated errors, and methods of calibration. A method for PMU-based remote calibration proposed by Ming Zhou[12] is explored. Since this calibration does not compensate for bad data or potential communication errors, a method for applying data verification methods to the installed PMUs is discussed as well.

3.2 Instrument Transformers

As transmission voltages and currents are far too high to measure with standard instrumentation, a transformer is required between the power system element and the connected measurement device. There are two types of instrument transformers, voltage or potential transformers (PTs), and current transformers (CTs). Both of these reduce the full-scale quantities to something that an instrument such as a PMU can handle safely and conveniently. In the United States, the secondary (low) values on these transformers are generally 120 volts (line-line) and 5 amperes.

3.2.1 Instrument Transformer Errors

An ideal instrument transformer would mirror the exact magnitude (proportional to the turns ratio) and phase of the full-scale primary signal. Unfortunately, real-world transformers do not behave in this fashion. Instead, instrument transformers have both a ratio correction factor (RCF) and phase angle correction factor (PACF) which must be determined for the transformer

to be calibrated correctly. As has been emphasized previously, the calibration of these two correction factors is required for PMU-based computational isolation, and hence, is critical to the implementation of reconfigurable monitoring.

The ubiquity of instrument transformers within a power system requires that a set of standards be followed regarding the size and performance of these devices. In the United States, IEEE C57.13, "Requirements for Instrument Transformers," is typically relied upon[13]. The standard defines three distinct accuracy classes for *metering* instrument transformers - 0.3, 0.6, and 1.2. These are essentially the maximum percentage error, in terms of the ratio. The table below outlines the limits for each accuracy class for instrument transformers within an expected operating range.

Table 3.1 Accuracy Class for Metering Service and Corresponding Limits of RCF

Metering Accuracy Class	**Voltage Transformers**		**Current Transformers**			
	90-100% Rated Voltage		*10% Rated Current*		*100% Rated Current*	
	Minimum	Maximum	Minimum	Maximum	Minimum	Maximum
0.3	0.997	1.003	0.994	1.006	0.997	1.003
0.6	0.994	1.006	0.988	1.012	0.994	1.006
1.2	0.988	1.012	0.976	1.024	0.988	1.012

The errors on metering voltage transformers are straightforward - the maximum percentage error directly corresponds to the accuracy class if the system is operating within expected limits. The errors on current transformers, however, are not so straightforward. With metering CTs, the line current must be compared against the rated current. Essentially, the further from rated current, the higher the error on the transformer is. For both PTs and CTs of any given accuracy class, the respective RCF and PACF limits form a parallelogram where the operating point(s) exist. Figures 3.1 and 3.2, taken from the IEEE standard c57.13[13], shows these operating regions for the 0.3, 0.6, and 1.2 metering accuracy classes.

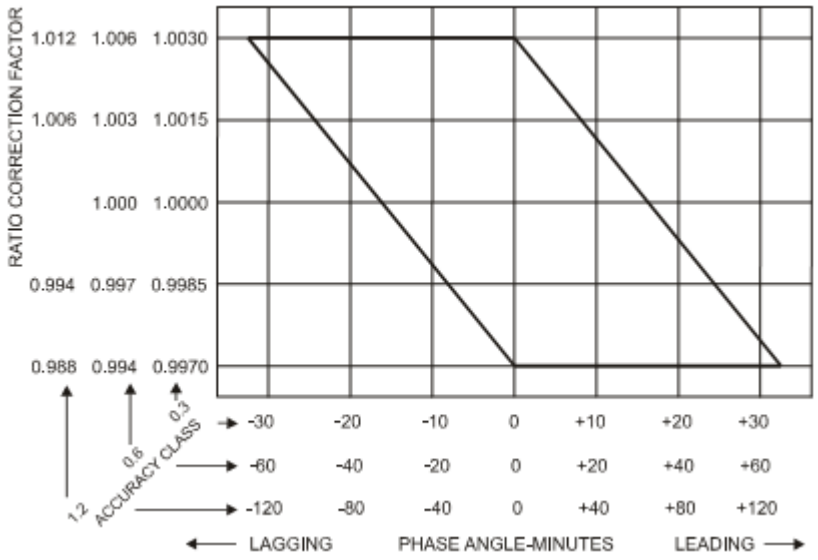

Figure 3.1: Limits of Accuracy Classes for Potential Transformers for Metering

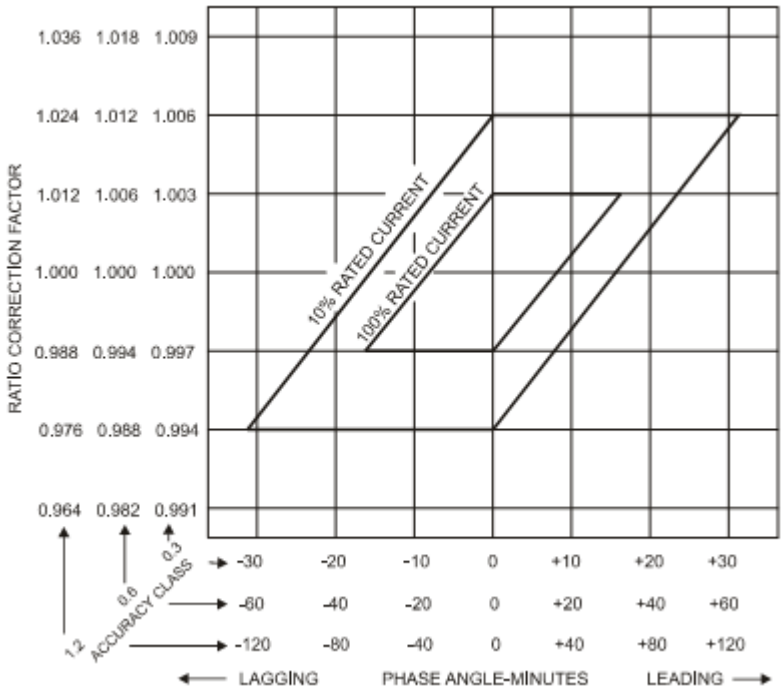

Figure 3.2: Limits of Accuracy Classes for Current Transformers for Metering

For a phasor measurement unit to function effectively as a border, the operating point (RCF, PACF) within these PMUs must be known so that the synchrophasor reading can be

appropriately adjusted. With adequate measurement redundancy in the system, these operating points can be determined without the need for calibration of instrumentation transformers in the field.

3.2.2 Instrument Transformer Calibration Using Phasor Measurements

To determine the RCF and PACF of instrumentation transformers through the use of synchrophasor measurements, an understanding of the underlying equations is first required. For voltage transformers, given the true phasor measurement and associated quantization error, $E_0 + \varepsilon_E$, the modified voltage phasor measurement can be written as:

$$E = E_0 \times (a + jb) + \varepsilon_E \tag{3.1}$$

where $a + jb$ represents the complex voltage transformer correction factors which must be determined. Conversion of this formula from polar to rectangular coordinates yields the following linear equations:

$$\begin{bmatrix} E_r \\ E_i \end{bmatrix} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \times \begin{bmatrix} E_{0r} \\ E_{0i} \end{bmatrix} + \begin{bmatrix} \varepsilon_{Er} \\ \varepsilon_{Ei} \end{bmatrix} \tag{3.2}$$

where subscripts *r* and *i* represent the real and imaginary parts of the measurement, respectively. The modified current phasor measurement equation is of the same form as the voltage phasor expressed in (3.1):

$$I = I_0 \times (c + jd) + \varepsilon_I \tag{3.3}$$

where $c + jd$ represents complex PT and CT correction factors to be determined.

As current phasors may be calculated from system state, the true phasor measurement I_0 has the following relation to the true voltage phasor measurement:

$$I_0 = Y \times E_0 \tag{3.4}$$

Converting from polar to rectangular yields:

$$\begin{bmatrix} I_{0(pq)r} \\ I_{0(pq)i} \end{bmatrix} = \begin{bmatrix} Y_r & -Y_i \\ Y_i & Y_r \end{bmatrix} \times \begin{bmatrix} E_{0r} \\ E_{0i} \end{bmatrix} \tag{3.5}$$

Substituting (3.5) into (3.3) and expanding, the resulting equations for the current phasor measurements are:

$$\begin{bmatrix} I_{(pq)r} \\ I_{(pq)i} \end{bmatrix} = \begin{bmatrix} c & -d \\ d & c \end{bmatrix} \times \begin{bmatrix} Y_r & -Y_i \\ Y_i & Y_r \end{bmatrix} \times \begin{bmatrix} E_{0r} \\ E_{0i} \end{bmatrix} + \begin{bmatrix} \varepsilon_{Ir} \\ \varepsilon_{Ii} \end{bmatrix} \tag{3.6}$$

where *r* and *i* denote the real and imaginary parts, respectively, and Y represents the admittance matrix composed of line and bus conductances and susceptances.

The equations given above describe how the RCF and PACF relate to the phasor measurements, and hence, to the state variables. Therefore, these correction factors can be converted from model parameters into independent state variables and solved in conjunction with the system state estimator. The caveat is that if sufficient measurement redundancy is not available, this results in having many more unknowns than equations, leading to an underdetermined system.

Since measurement redundancy cannot be guaranteed at every point where a PMU exists (as from an observability perspective, that may defeat the purpose of PMU placement), then another method must be used to ensure that the state estimator is not underdetermined.

Fortunately, under normal system operating conditions, instrument transformers function in the linear portion of their characteristic curves[14], meaning that the correction factors are predictable and repeatable. Essentially, despite changes in system state, instrument transformers maintain the same operating point under different operating conditions. To take advantage of this fact, Ming Zhou proposed a reformulation of the traditional state estimator in [12] which utilizes multiple scans of different system conditions and uses them to solve for the RCF and PACF for instrument transformers in the system.

This method requires an augmented Jacobian matrix H, which includes partial derivatives with respect to the new independent state variables, the PT and CT RCFs and PACFs:

$$H = \begin{bmatrix} \frac{\partial E_r}{\partial E_{0r}} & \frac{\partial E_r}{\partial E_{0i}} & \frac{\partial E_r}{\partial a} & \frac{\partial E_r}{\partial b} & \frac{\partial E_r}{\partial c} & \frac{\partial E_r}{\partial d} \\ \frac{\partial I_{(pq)r}}{\partial E_{0r}} & \frac{\partial I_{(pq)r}}{\partial E_{0i}} & \frac{\partial I_{(pq)r}}{\partial a} & \frac{\partial I_{(pq)r}}{\partial b} & \frac{\partial I_{(pq)r}}{\partial c} & \frac{\partial I_{(pq)r}}{\partial d} \\ \frac{\partial E_i}{\partial E_{0r}} & \frac{\partial E_i}{\partial E_{0i}} & \frac{\partial E_i}{\partial a} & \frac{\partial E_i}{\partial b} & \frac{\partial E_i}{\partial c} & \frac{\partial E_i}{\partial d} \\ \frac{\partial I_{(pq)i}}{\partial E_{0r}} & \frac{\partial I_{(pq)i}}{\partial E_{0i}} & \frac{\partial I_{(pq)i}}{\partial a} & \frac{\partial I_{(pq)i}}{\partial b} & \frac{\partial I_{(pq)i}}{\partial c} & \frac{\partial I_{(pq)i}}{\partial d} \end{bmatrix} \tag{3.7}$$

$$= \begin{bmatrix} a & -b & E_{0r} & -E_{0i} & 0 & 0 \\ cY_r - dY_i & -(cY_i + dY_r) & 0 & 0 & Y_r E_{0r} - Y_i E_{0i} & -(Y_i E_{0r} + Y_r E_{0i}) \\ b & a & E_{0i} & E_{0r} & 0 & 0 \\ cY_i + dY_r & cY_r - dY_i & 0 & 0 & Y_i E_{0r} + Y_r E_{0i} & Y_r E_{0r} - Y_i E_{0i} \end{bmatrix}$$

Given a large enough set of system scans, it is possible to solve for multiple instances of the system state and a single instance of the RCF and PACF values (a, b, c, d) for the instrument transformers. The formulation of this time-lapse state estimation is presented on the following page.

Let

$$T_n = \begin{bmatrix} \frac{\partial E_r}{\partial E_{0r}}(E_n) & \frac{\partial E_r}{\partial E_{0i}}(E_n) \\ \frac{\partial I_{(pq)r}}{\partial E_{0r}}(E_n) & \frac{\partial I_{(pq)r}}{\partial E_{0i}}(E_n) \\ \frac{\partial E_i}{\partial E_{0r}}(E_n) & \frac{\partial E_i}{\partial E_{0i}}(E_n) \\ \frac{\partial I_{(pq)i}}{\partial E_{0r}}(E_n) & \frac{\partial I_{(pq)i}}{\partial E_{0i}}(E_n) \end{bmatrix}, \; F_n = \begin{bmatrix} \frac{\partial E_r}{\partial a}(E_n) & \frac{\partial E_r}{\partial b}(E_n) & \frac{\partial E_r}{\partial c}(E_n) & \frac{\partial E_r}{\partial d}(E_n) \\ \frac{\partial I_{(pq)r}}{\partial a}(E_n) & \frac{\partial I_{(pq)r}}{\partial b}(E_n) & \frac{\partial I_{(pq)r}}{\partial c}(E_n) & \frac{\partial I_{(pq)r}}{\partial d}(E_n) \\ \frac{\partial E_i}{\partial a}(E_n) & \frac{\partial E_i}{\partial b}(E_n) & \frac{\partial E_i}{\partial c}(E_n) & \frac{\partial E_i}{\partial d}(E_n) \\ \frac{\partial I_{(pq)i}}{\partial a}(E_n) & \frac{\partial I_{(pq)i}}{\partial b}(E_n) & \frac{\partial I_{(pq)i}}{\partial c}(E_n) & \frac{\partial I_{(pq)i}}{\partial d}(E_n) \end{bmatrix}$$

where T_n represents the Jacobian components for the system state variables for a given scan, and F_n represent the Jacobian components for the RCF and PACF for a given scan. The augmented state estimation matrices are then constructed as:

$$H_{aug} = \begin{bmatrix} T_1 & & & F_1 \\ & T_2 & & F_2 \\ & & \ddots & \vdots \\ & & T_n & F_n \end{bmatrix}, \; W_{aug} = diag(W_1, W_2, \dots W_n), \; \Delta z_{aug} = \begin{bmatrix} \Delta z_1 \\ \Delta z_2 \\ \vdots \\ \Delta z_n \end{bmatrix} \tag{3.10}$$

where Δz represents the difference between the estimated and measured values for a given iteration of a given measurement set. Each iteration solves for the state variables as a traditional state estimator would:

$$\begin{bmatrix} \Delta E_{r1} \\ \Delta E_{i1} \\ \vdots \\ \Delta E_{rn} \\ \Delta E_{in} \\ \Delta a \\ \Delta b \\ \Delta c \\ \Delta d \end{bmatrix} = \left[H_{aug}^T W_{aug}^{-1} H_{aug}\right]^{-1} H_{aug}^T W_{aug}^{-1} \Delta z_{aug} \tag{3.11}$$

This iterative equation is an augmented form of (2.5), and the state estimator iterates until the objective function $J(x) = \Delta z_{aug}^T W_{aug} \Delta z_{aug}$ reaches a desired minimum value, as with a

traditional state estimator. This reformulation may be used with system-wide state measurement sets, or with subsets created by computational islanding. The key is that multiple scans are used to solve the RCF and PACF; however, as this process is otherwise identical to a traditional state estimator, computational borders established with PMUs may be used without any further reformulation.

3.3 Bad Data Detection in State Estimation

Although state estimation functions in a computationally islanded system just as it would in a non-islanded one, there are some extra considerations with regards to common problems. In particular, is the question of how to handle bad data or faulty measurements. Methods of detecting bad data are employed in nearly all state estimators[6], and are typically used in a pre-processing step before the measurement set is established for a given execution of the estimator.

The exact statistical method used in bad data detection varies by implementation, although a chi-square test or analysis of the standard deviation is common. Regardless of the method, bad data identification can be performed on the regions formed with computational islanding. Unfortunately, the synchrophasor data does not have such a check directly in the state estimator, as the PMU-enabled buses are not included as part of the measurement set - they are not calculated as part of the state estimate, only measured. Even if the instrumentation transformers are properly calibrated with a method such as the one presented in section 3.2.2, a PMU hardware malfunction or persistent communication error is still a possibility. Performing a bad data detection test on the synchrophasor measurements then requires an estimate of the state variable which is determined through alternate means. If measurement redundancy is not available, then this is not a possibility. In such a case, the synchrophasor measurement is considered to be a critical measurement. The term *critical*, as applied to a measurement or group

of redundant measurements (k-tuple), implies that bad data detection is not possible and the state estimator is underdetermined if these measurements are lost. Furthermore, in the case of a critical k-tuple, methods of bad data detection can only verify that one or more of the measurements is in error, but the specific measurement cannot be determined[6].

Typically, as synchrophasors are more accurate than conventional power system measurements, it has been suggested that PMUs serve as redundant measurements in a power system[15]. However, this redundancy works in both directions. Given a non-critical synchrophasor measurement, a bad data test can be performed by comparing the synchrophasor to a traditional estimate. The simple method for doing this is to use the state variable estimate determined by a full system state estimation. Alternatively, the computational islands can be reconfigured such that the PMU in question is no longer trusted, and a larger island is created. The abstract system below illustrates this method:

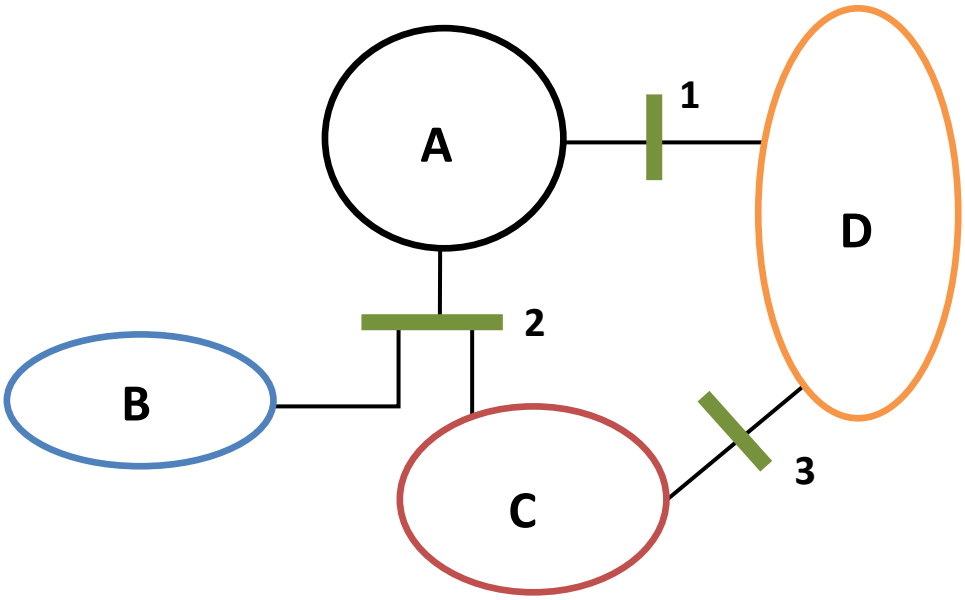

Figure 3.3: Abstract System with PMU-Established Computational Islanding

To validate the measurements provided by the border PMU on bus 2, then the measurements initially treated as inaccurate, and the islands are aggregated together. This results in the following islanding configuration:

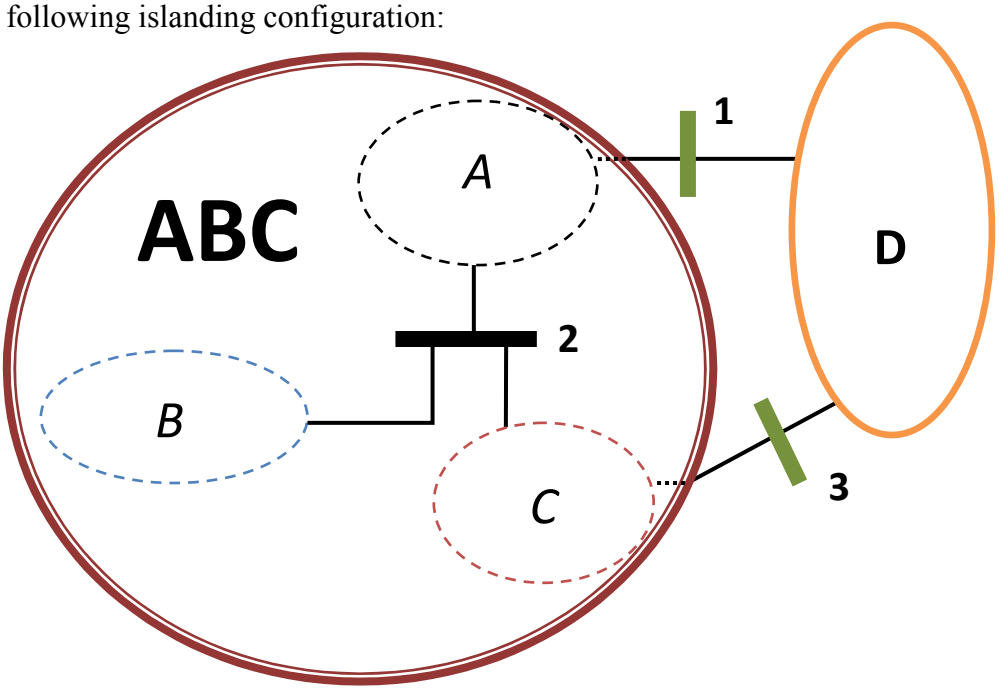

Figure 3.4: Abstract System with Computational Islanding for PMU Data Validation

It should be noted that this regrouping will result in a longer computation time for the aggregated island. However, this same technique may be used to ensure that computational boundaries mesh closely with electrical ones. Given the above system, any of the following computational islanding configurations could be used at the discretion of the system operator:

Table 3.2 Computational Islanding Configurations for an Abstract System

PMUs Used as Borders	PMUs to Validate	Resulting Computational Islands
None	1,2,3	1: ABCD
1	2,3	1: ABCD
2	1,3	2: B, ACD
3	1,2	1: ABCD
1,2	3	3: A, B, CD
1,3	2	2: ABC, D
2,3	1	3: B, C, AD
1,2,3	None	4: A, B, C, D

Three of the possible combinations actually result in no meaningful computational islanding - the entire system must be calculated. However, as reconfigurable monitoring is not intended to replace traditional state estimation, data from a full system estimation should be available to the system operators in most cases. Although the PMU data can never be guaranteed to be completely accurate, the system must be resilient enough to allow for statistical methods to be utilized and for bad or faulty data points to be removed.

To summarize, utilizing PMUs as computational borders requires extra computational effort in terms of both instrument transformer calibration and bad data detection. This is not to compensate for accuracy issues in the PMU, but rather the supporting hardware, for both measurement and communication. However, neither of these methods impose an undue hardware burden in a typical system - the methods are purely computational, although measurement redundancy is required. This ensures that the synchrophasor measurement data is as accurate and error-resistant as possible, which is crucial for the implementation of a PMU-based computational islanding scheme.

Chapter 4. PMU Placement Algorithm

4.1 Introduction

The previous chapters have addressed the theory behind reconfigurable monitoring - PMUs as borders within a state estimator and the practical considerations that must be handled. However, the strategic placement of PMUs within a power system has not been discussed. This chapter provides an overview of PMU placement algorithms and system models, and presents an algorithm which is designed to maximize the number of computational islands in a system. This, in effect, maximizes the potential benefit of reconfigurable monitoring. As with other placement algorithms, a consistent system model must be established. The rules for creating placement models are explained and justified in this chapter. Following that, the placement algorithm is presented along with an example PMU placement cycle.

4.2 Placement Model

The PMU placement problem is not a new one - many algorithms have been developed to determine optimal placement with regards to observability. These algorithms typically either rely on graph theory or integer programming to place the PMUs, although in both cases a placement model is created. The concept behind placement models is that a given application may not need the details of a full planning model, such as the models which are used in PSS/E[16]. Instead, abstractions may be made - for PMU observability placement, a bus/branch model is typically sufficient. In these models, the power system is essentially reduced to a series of nodes and vertices, which is why graph theory concepts are often used in these sorts of placement problems. Beyond simple abstraction, various rules can be used to simplify the system by removing superfluous buses or branches - elements of the planning model which may be "virtual" in nature but cannot support instrumentation. While this can help with computation

time, it is more useful for eliminating impossible or unnecessary placement candidates. Given the goal of the placement algorithm presented here - to maximize the number of computationally isolated regions within a system - the following rules are used to simplify the system.

4.2.1 Placement Model Reduction Rules

The rules presented in this section are essentially a subset of the rules used for an observability algorithm placement model. Credit for the formalization of these rules goes to [12] and [17].

Transformers:

From a system planning perspective, each side of a transformer may be considered as a separate bus. For state estimation and wide area monitoring, however, the transformer buses may be abstracted into a single bus - typically the high voltage - and the turns ratio and configuration can be used to determine the voltage on the low side.

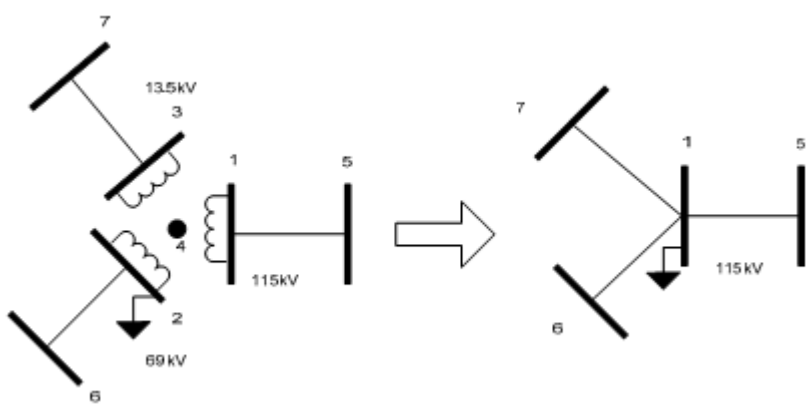

Figure 4.1: Transformer Reduction (3-Winding)

DC Transmission Lines:

While the power carried by DC transmission lines must be measured, the DC buses themselves do not play a role in state estimation. Furthermore, there is no need for synchrophasor measurements to measure DC voltage or current, so they are not considered in a PMU placement model. Instead, the AC buses which interface with the DC buses remain in the model, and the DC power is treated as an injection on the AC bus. In a sense, DC links are able perform the same computational islanding that a PMU is capable of.

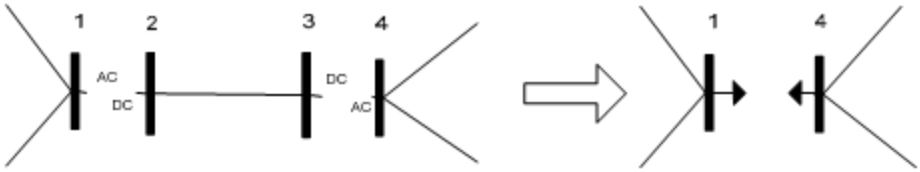

Figure 4.2: DC Transmission Line Reduction

Tapped Transmission Line:

As mentioned previously, not every bus in a planning model corresponds to a real bus where instrumentation may be installed. For a tapped transmission line, a PMU installation is not possible. Therefore, when considering potential placement locations, the bus representing the tap should be removed from consideration and a power injection measurement added to each end bus While this also has the effect of creating a computational border, it is justifiable as the state variables at the tap are not required - the stability and operating condition of the system can be determined by examination of the other buses connected to the tap.

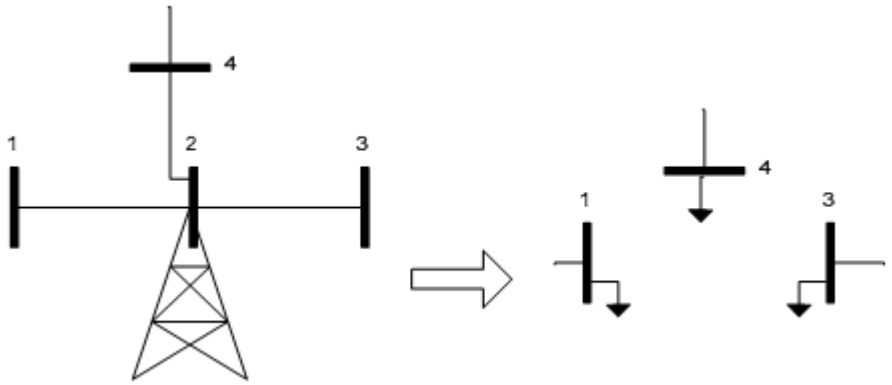

Figure 4.3: Tapped Transmission Line Reduction

Series Capacitor:

In a planning model, a series capacitor may be connected to buses which, as with a tapped line, do not actually exist in the physical system. No PMU can be placed on these buses, nor are these state variables necessary for monitoring the system. The capacitive reactance can be added to the appropriate branch in the admittance (Y) matrix, and the capacitor and buses may be removed. This has no effect on computational islanding potential, as any islanding which could be performed with a PMU on the capacitor's buses can also be performed with the remaining end buses.

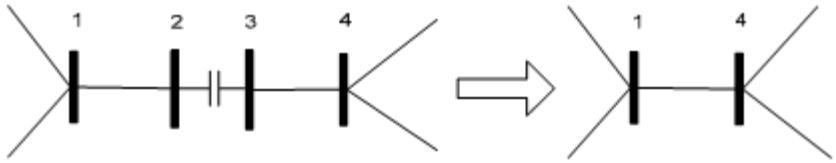

Figure 4.4: Series Capacitor Reduction

Dummy/False Buses and Multi-section Lines:

In a planning model, buses may exist solely for simulation purposes - these buses may allow for the placement of simulated faults or metering equipment within the planning software.

As these buses do not actually exist in the system, a PMU may not be installed at these locations, and so they should be removed from the placement model. A false bus actually exists within the power system, but is essentially a zero injection bus connected with only two branches. This is generally part of a multi-section line, and as with the series capacitor, only the buses on either end must be considered for PMU placement. The state variables of these extra buses are once again not required for system monitoring, and any computational islanding that can be performed with a placement on a false bus can also be achieved with a placement on one of the end buses.

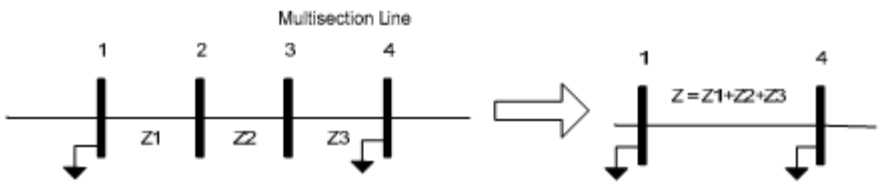

Figure 4.5: Multi-section Line/Dummy Bus/False Bus Reduction

In addition to the rules which have been outlined, solitary buses in the planning model which are isolated from the rest of the power system may be removed, although discretion must be exercised in doing so. If the buses represent a portion of the system that is being constructed, then long-term planning may warrant their inclusion in the PMU placement model, although the benefits will not be immediately seen. Shunt elements and generators which are connected to their own buses may be aggregated and abstracted onto a larger substation bus. This aggregation is often performed for the purposes of state estimation; however, as an aggregated form is a non-trivial model change and is not always used, such reductions are not explicitly considered as part of these placement rules.

4.3 Placement Algorithm Overview

On a high level, the computational islanding algorithm functions very much like an observability algorithm. The topology of the system is processed, and PMUs are placed according to a set of criteria. The simplest way of placing PMUs is with a greedy algorithm[17]. In this case, the greedy algorithm has three inputs: the system topology, the desired number of PMUs, and the number of PMUs already in the system (and their locations). A small initial set of PMUs helps to direct the algorithm, but is not required. This allows for the solution to be constructed in incremental steps, or for a solution to be constructed from a pre-placed set of PMUs. For example, PMUs may already be placed in a system for the purpose of observability, and while no new PMUs are being added, the algorithm can analyze the placement locations and create as many computational islands as possible.

In the design of this specific algorithm, there are three basic goals: a) maximization of computational islanding, b) consideration of PMUs which already exist in the system, and c) determination of a placement order. This results in a solution which is optimal for a growing power system and ensures that the potential for reconfigurable monitoring is maximized while keeping up-front hardware investment as low as possible. A high-level flowchart of the process is presented on the next page.

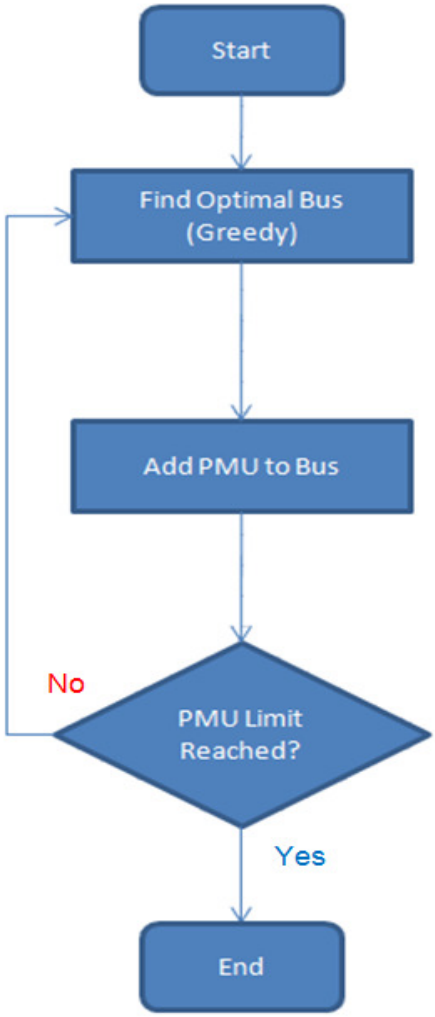

Figure 4.6: High Level-PMU Placement Algorithm Flowchart

The above flowchart assumes that the system has already been reduced and that the list of buses in consideration is a minimal set. This is a purely greedy algorithm, and as with algorithms designed for placement with respect to observability, there is no guarantee that this is necessarily the best solution. However, it does require much less computing power than a combinatorial or heuristic solution, and consideration of a single PMU at a time allows for smooth future expansion of the placement set.

4.4 Placement Algorithm Details

The placement algorithm takes two things into consideration - the number of islands created when a PMU is placed, and the bus voltage at the placement location. If the number of islands is the same, then the higher voltage bus is the placement target. In the event of a tie, the first location is chosen by default. This tiebreaker method is not necessarily any more effective than choosing at random, but ensures a consistent placement. The island count itself is performed with a depth first search. The process is iterative, and a rough estimate of the computational complexity is given by:

$$Complexity = N_{PMU} \times N_{Bus} \times (N_{Bus} - N_{Terminal})$$

where N_{PMU} is the total number of PMUs to be placed, N_{Bus} is the number of buses in the system, and $N_{Terminal}$ is the number of buses with only a single branch. These terminal buses are not considered for PMU placement for this method, as a PMU placed on a terminal bus would not result in the creation of a computational island. The depth first search itself is recursive, and the exit conditions are as follows:

- *Does the bus have a PMU present?*
- *Has this bus been visited before?*

By using these exit criteria, combined with the implicit terminal case (the depth first search will return if there is nowhere else to go), the islands in the system can be counted. The depth first search algorithm is run on every bus with a PMU, and a return to that initial PMU signifies the presence of an island which encapsulates all buses encountered on the traversal. To more easily illustrate this, a detailed flowchart for the algorithm as well as a computational islanding solution for the IEEE 14 bus test system are presented.

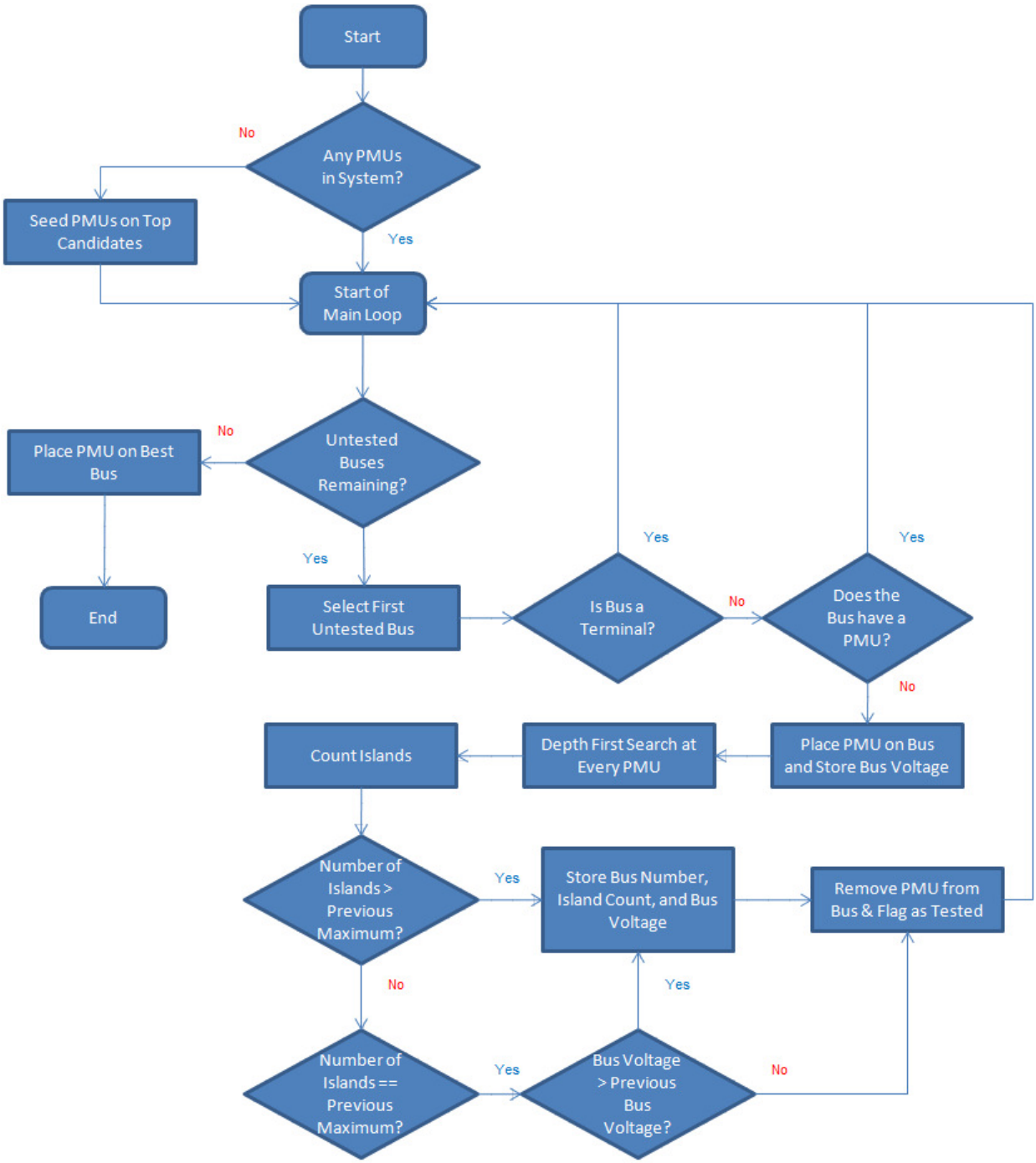

Figure 4.7: Greedy Computational Islanding Algorithm Flowchart

As illustrated in Figure 4.7, if there are no PMUs in the system, then some are placed at random on the best potential candidates - high voltage buses with many branches. In the main

loop, the greedy algorithm places a temporary PMU on a bus and then executes a depth first search starting at every PMU currently in the system. The primary decision in PMU placement is the number of computational islands created, although in the case of a tie, the bus with higher voltage is selected. If the voltages are equal, then the senior bus - the first one encountered by the algorithm - is selected. After all buses have been tested, the best candidate is selected. If there are still PMUs available to be placed, the cycle repeats until the PMU limit is reached. The implicit assumption here is that adjacent buses will be observable - either by branch current measurements, or other measurements already present in the system. While there is no requirement for the PMU to measure the currents on all branches connected to the installation bus, there is high practical incentive to do so. The only requirement on the system is that the set of measurements must be able to provide complete observability so that all state variables can be solved.

To illustrate how a depth first search is capable of determining the computational islands in a power system, a single execution on the IEEE 14 bus test system is performed. A PMU is initially seeded on bus 6, and PMU placement buses are tested in ascending order (1-14). A table summarizing the results is given below and a diagram outlining the depth first search execution is presented on the following page.

Table 4.1 Computational Islanding Algorithm Results - 14 Bus Test System

Test Placement Bus	Computational Islands	Test Placement Bus	Computational Islands
1	1	*8*	No Count - Terminal
2	1	*9*	3
3	1	*10*	2
4	2	*11*	1
5	1	*12*	1
6	No Count - PMU	*13*	2
7	2	*14*	2

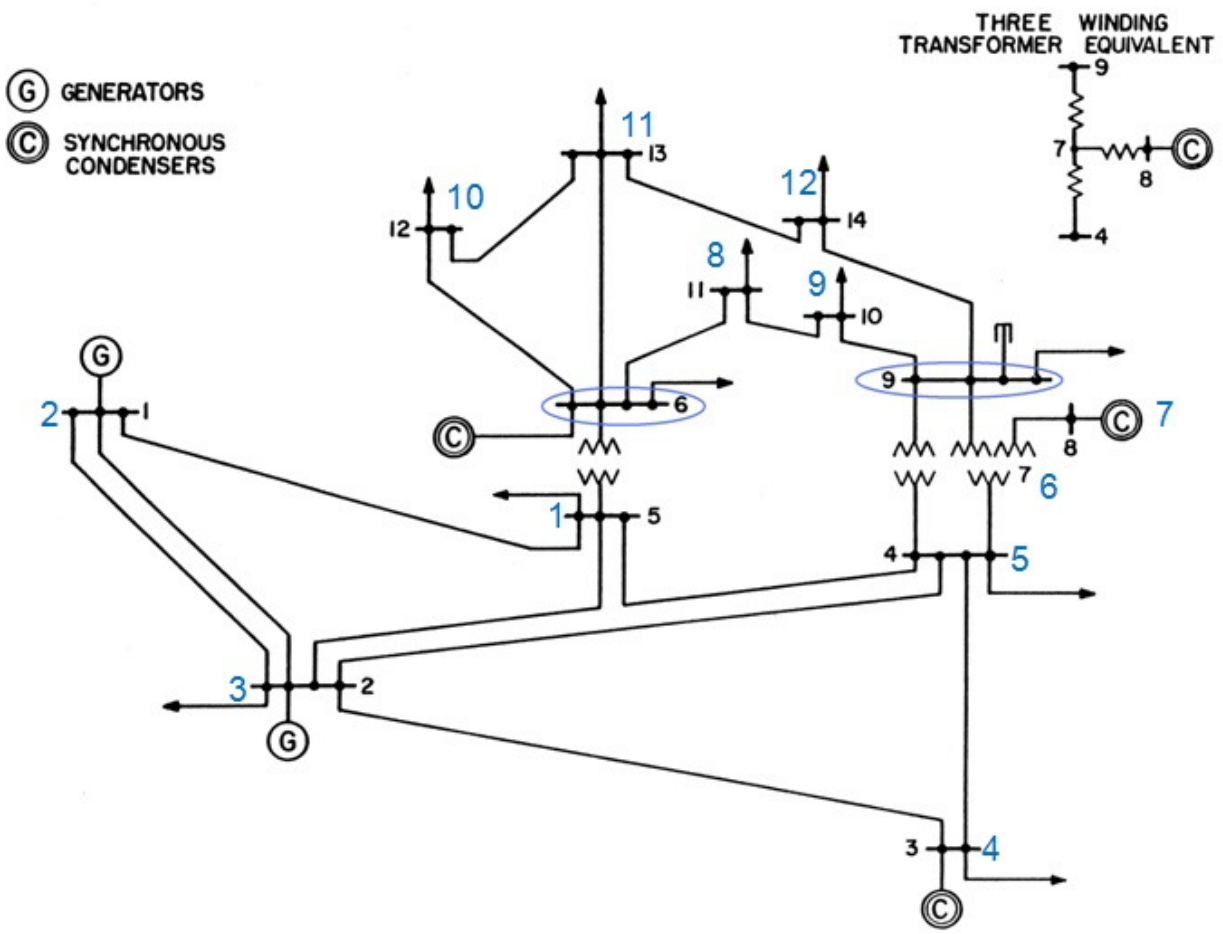

Figure 4.8: Depth First Search System Traversal - Low Bus Number Priority

Figure 4.8 depicts the order in which new buses are discovered on a given traversal. When a new bus is reached, the depth first traversal continues starting with the lowest numbered adjacent bus. The traversal order (and logic) is as follows:

6 (*Start - Island 1*), 5, 1, 2, 3, 4, 7, 8, 9 (*PMU*); Return - All Buses Visited

6 (*Start - Island 2*), 11, 10, 9 (*PMU*); Return - All Buses Visited

6 (*Start - Island 3*), 12, 13, 14, 9 (*PMU*); Return - All Buses Visited

The backtrace steps are omitted in this traversal - in each case, once the traversal reaches bus 9, every other bus in the island has been discovered. This is not necessarily the case for every

potential traversal, and only occurs here due to the system's internal ordering of the buses. To verify that the islanding works regardless of the internal ordering, the following diagram presents the traversal when the highest numbered adjacent bus takes priority:

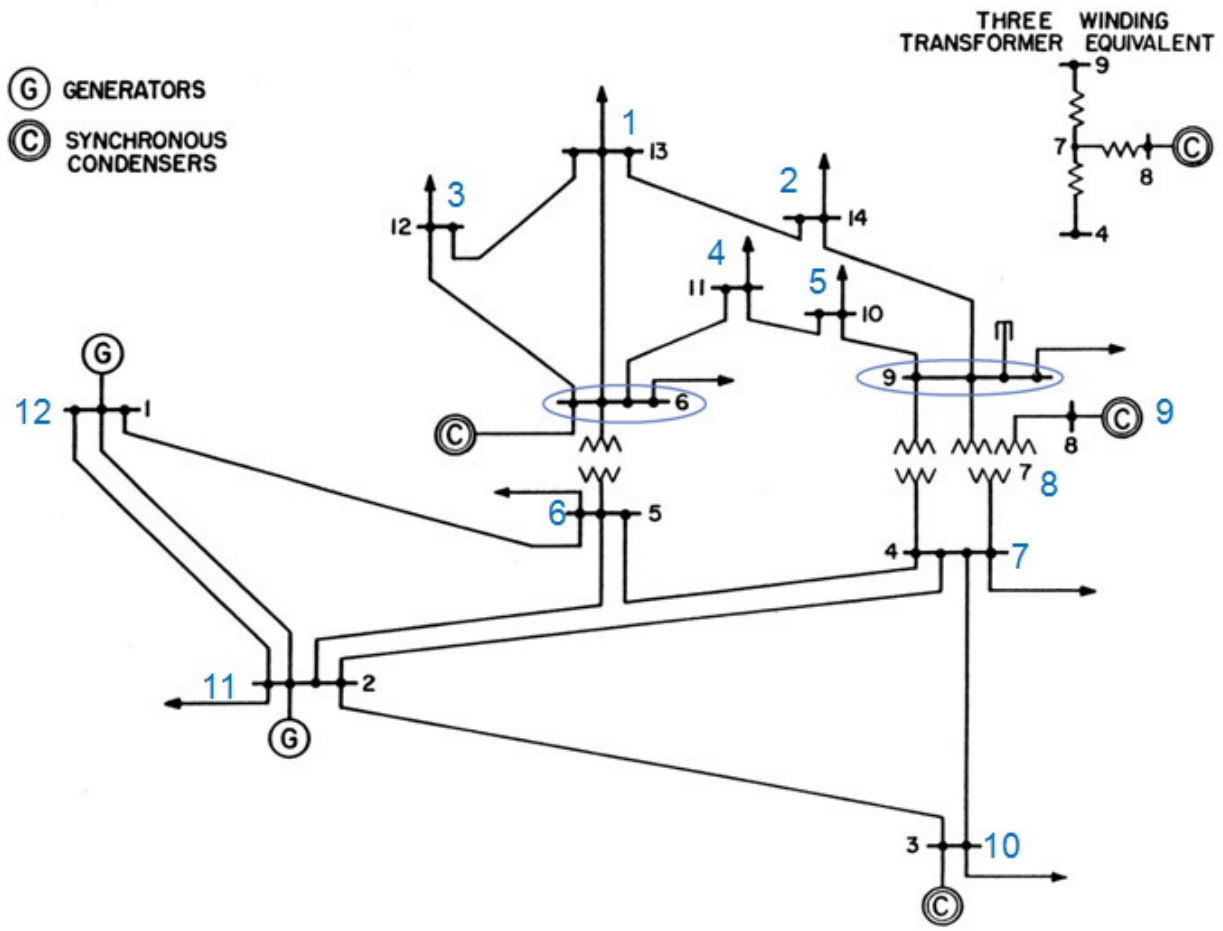

Figure 4.9: Depth First Search System Traversal - High Bus Number Priority

The traversal logic for the system in this case is:

6 (*Start - Island 1*), 13, 14, 9 (*PMU*), 12; Return - All Buses Visited

6 (*Start - Island 2*), 11, 10, 9 (*PMU*); Return - All Buses Visited

6 (*Start - Island 3*), 5, 4, 9 (*PMU*), 7, 8, 3, 2, 1; Return - All Buses Visited

Once again, backtrace steps have been omitted in the traversal. The same islands are created in either case; however, the latter example serves to show how the PMU functions as a generic halting condition - the island is actually created when the traversal is forced to return to the starting PMU and for a small system, such as the one presented, removing the buses which the PMUs are on and performing a simple visual inspection reveals the computational islands present in the system. As this quickly becomes impractical for larger systems, the thorough (albeit slow) depth first search algorithm is necessary to determine the layout of the computational islands. The MATLAB code for the algorithm is included in Appendix B.

Chapter 5. Applications

5.1 Introduction

This chapter presents examples of how reconfigurable monitoring can be used to enhance state estimation capabilities. Various computational islanding configurations are presented for 3 different systems - the IEEE 118 and 300 bus test systems, and a reduced 1457 bus Brazilian ONS transmission system. All of these configurations are the result of the PMU placement algorithm presented in Chapter 4, and as such are not necessarily optimal. However, the purpose is to illustrate the functionality that computational islanding provides to a state estimator. State estimation speed and accuracy tests are performed on the Brazilian ONS system, and state estimation fault tolerance is presented using the IEEE 300 bus test system.

5.2 Example Computational Islanding Configurations

The following islanding configurations are used for the 118 bus island reconfiguration and state estimation tests presented in sections 5.3 and 5.4:

Table 5.1 Computational Islanding Configurations

System	PMUs	Placement Buses	Computational Islands
IEEE 118 Bus	10	5, 12, 15, 30, 37, 49, 68, 77, 80, 100	17
IEEE 300 Bus	15	3, 23, 49, 59, 64, 85, 97, 109, 116, 143, 172, 203, 223, 267, 270	35
ONS Transmission	125	18, 23, 27, 31, 38, 40, 42, 43, 44, 45, 51, 57, 58, 59, 60, 61, 65, 67, 71, 73, 76, 79, 80, 82, 84, 86, 87, 89, 92, 99, 105, 115, 124, 126, 129, 130, 136, 141, 143, 145, 165, 188, 194, 197, 208, 240, 245, 246, 250, 259, 262, 263, 264, 267, 271, 274, 294, 311, 312, 316, 317, 320, 326, 333, 341, 365, 366, 367, 371, 388, 391, 397, 486, 504, 505, 509, 510, 524, 528, 530, 533, 544, 549, 552, 560, 606, 614, 628, 640, 644, 676, 692, 784, 794, 807, 829, 850, 868, 871, 1087, 1094, 1119, 1163, 1179, 1184, 1216, 1217, 1218, 1233, 1239, 1246, 1247, 1248, 1259, 1289, 1324, 1331, 1347, 1368, 1389, 1399, 1405, 1419, 1428, 1439	294

For the 118 bus test system, the islanding placement is presented in the following diagram. As the system diagram is not planar, the subsystem outlines cannot be presented.

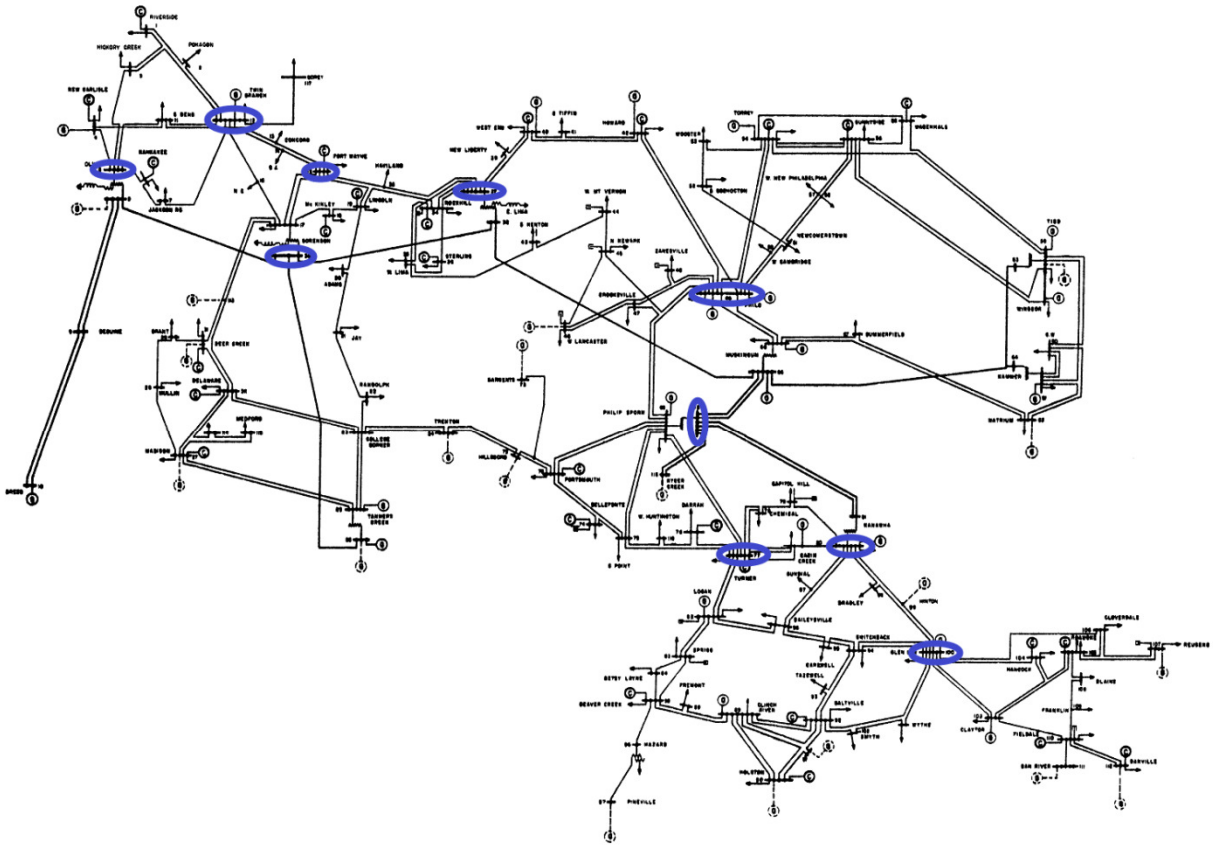

Figure 5.1: 118 Bus System PMU Placement - 17 Computational Islands

As this is a small system, 17 computational islands are not strictly required; a balance of flexibility and calibration requirements can be configured by aggregating some of the smaller islands. The diagram below illustrates the same PMU placement, but with only seven aggregated computational islands. This requires that only 6 PMUs have properly calibrated instrument transformers. It should be noted that this is only one possible configuration - the system state may justify a different reconfiguration based on the operating conditions.

For the aggregated islands, dark blue ellipses indicate PMUs used as borders, red ones indicate PMUs within the aggregate islands:

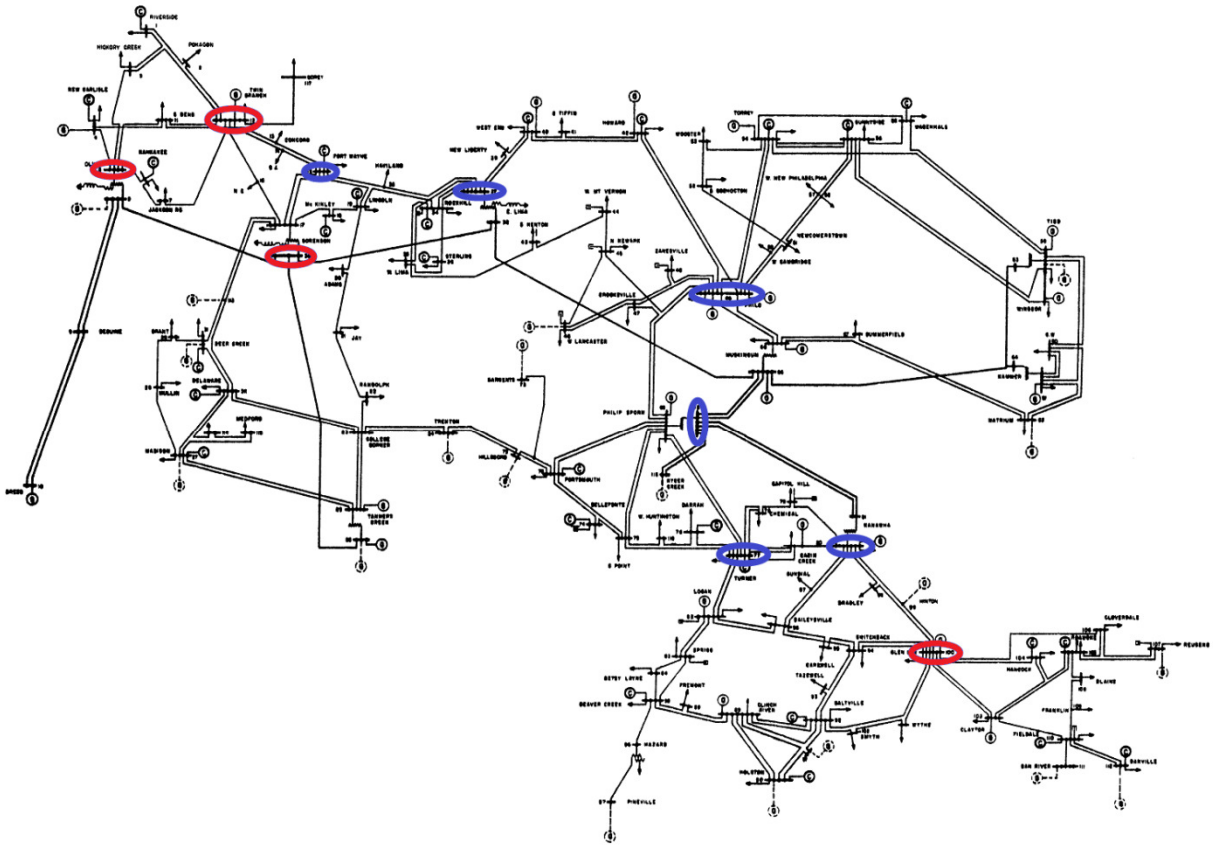

Figure 5.2: 118 Bus System PMU Placement - 7 Aggregated Computational Islands

5.3 Reconfigurable Monitoring Speed and Accuracy

Using the reduced Brazilian ONS transmission system model, tests are performed to compare the state estimation execution time of a full-system state estimator and an estimator with computational islands. The state estimation algorithm used is completely un-optimized; it is not PQ decoupled. However, the same algorithm is used in both cases, so that the only difference is in the method - full-system or split system.

These tests are performed on a computer with a 2.26 GHz Intel(R) Core(TM)2 Duo processor, 6 GB of PC3-8500 DDR3 RAM, and a 7200 RPM SATA 2.0 (3.0 Gb/s) hard drive. Three different load profiles are used, and each state estimation method is executed 50 times for each load profile - 150 executions for each method in total. All cases are injection-only; the measurement sets consist of power injections and voltage magnitudes at every non-PMU bus. PMUs are present in both systems, and their PTs are assumed to have been calibrated. The results are summarized in Tables 5.2 and 5.3 and presented visually in Figures 5.3-5.5.

Table 5.2 State Estimation Execution Speed Test Summary - Traditional State Estimator

	Average Runtime (s)	*Standard Deviation (s)*	*Median # Iterations*
Load Profile 1	137.31	0.172	5
Load Profile 2	139.82	0.201	5
Load Profile 3	161.18	0.171	6

Table 5.3 State Estimation Execution Speed Test Summary - Islanded State Estimator

	Average Runtime (s)	*Standard Deviation (s)*
Load Profile 1	6.52	0.085
Load Profile 2	6.48	0.091
Load Profile 3	6.77	0.105

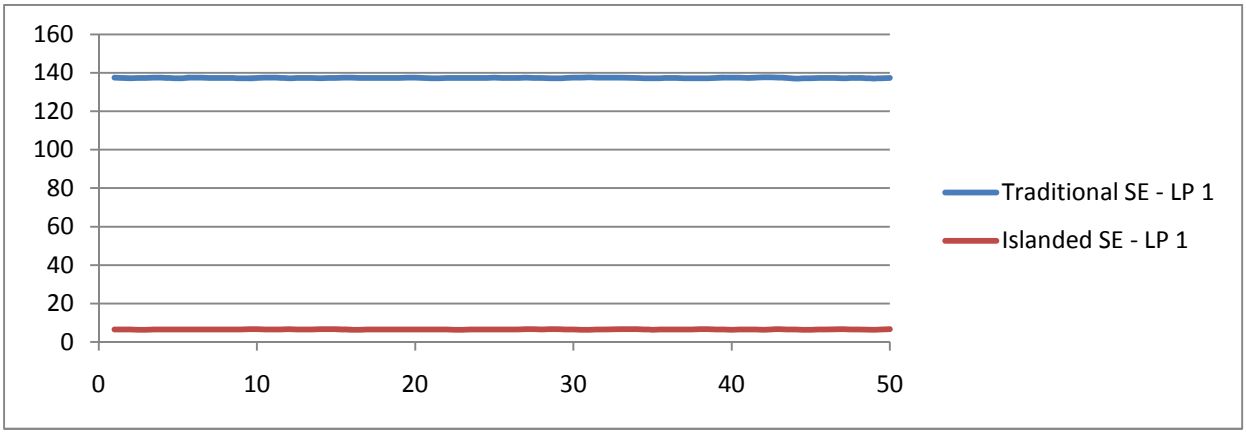

Figure 5.3: State Estimation Speed - Load Profile 1

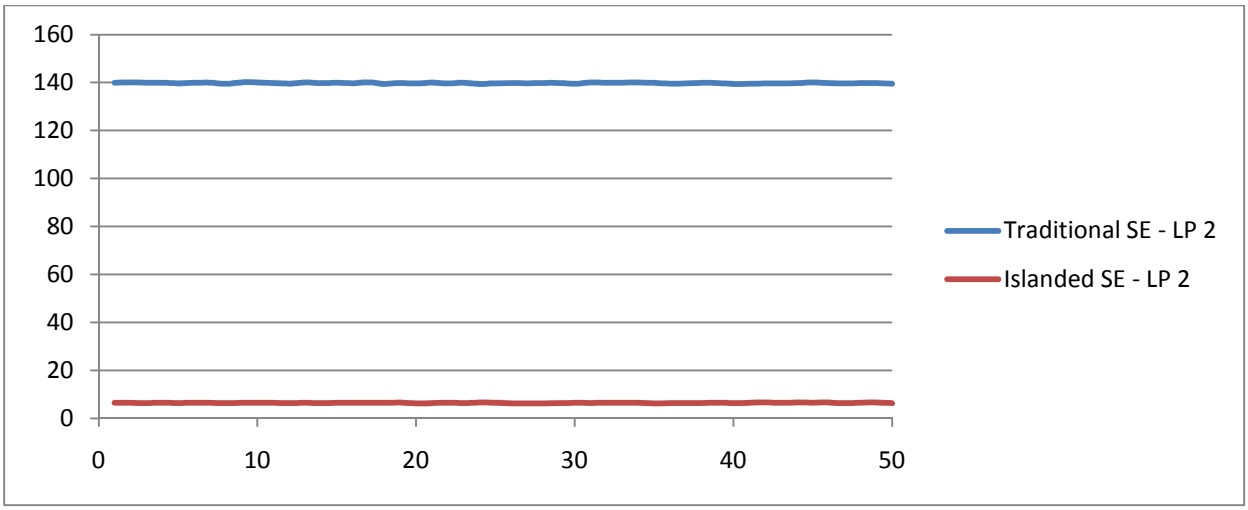

Figure 5.4: State Estimation Speed - Load Profile 2

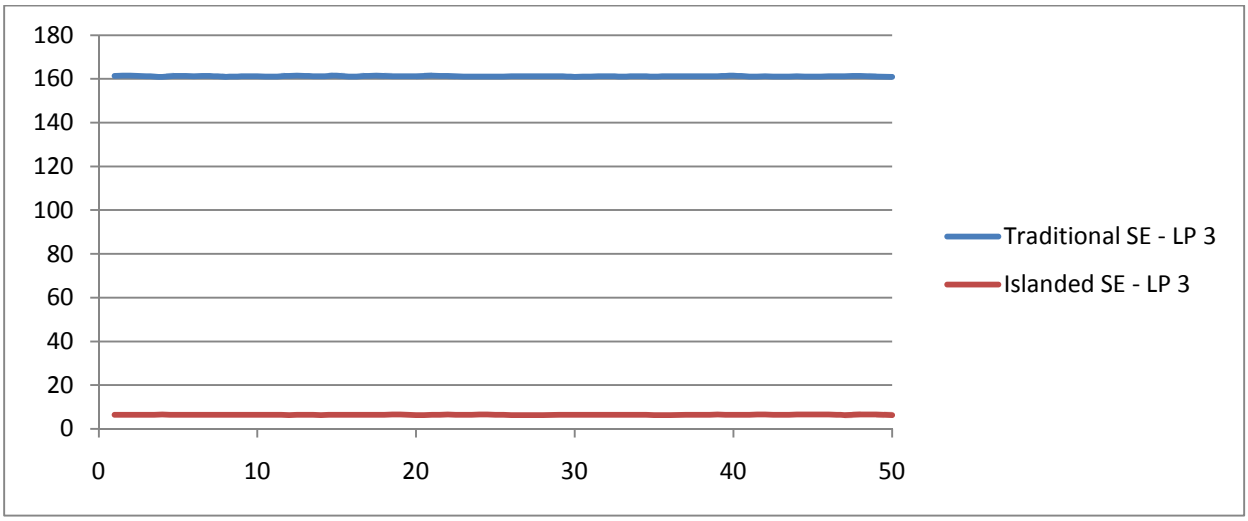

Figure 5.5: State Estimation Speed - Load Profile 3

In all cases, the state estimator with computational islands has a much faster runtime. It should be acknowledged that due to the increased computational overhead involved, the traditional method may benefit more from potential algorithm optimizations. The same tests are also used to determine the accuracy of the computationally islanded state estimator. The differences in the state estimates are summarized in Table 5.3. Figures 5.6-5.8 illustrate the total average difference for each execution for the three load profiles.

Table 5.4 Reconfigurable Monitoring Accuracy Test Summary

	Total # Differences > 0.001 (50 Estimations)	*Median # Differences*	*Average (Mean) Difference Value*	*Maximum Difference*
Load Profile 1	107 (0.0735%)	2	0.00217	0.007
Load Profile 2	152 (0.1044%)	3	0.00238	0.006
Load Profile 3	144 (0.0989%)	3	0.00188	0.004

To qualify any differences between the traditional estimator and the islanded estimator with respect to a reasonable precision, the state variable estimates are compared bus by bus. If there is a disagreement of 0.001 or greater for a given bus between the two methods, it is counted. The median number of disagreements is tallied, along with the average difference (using all 50 cases as the population), and the maximum difference from all 50 executions.

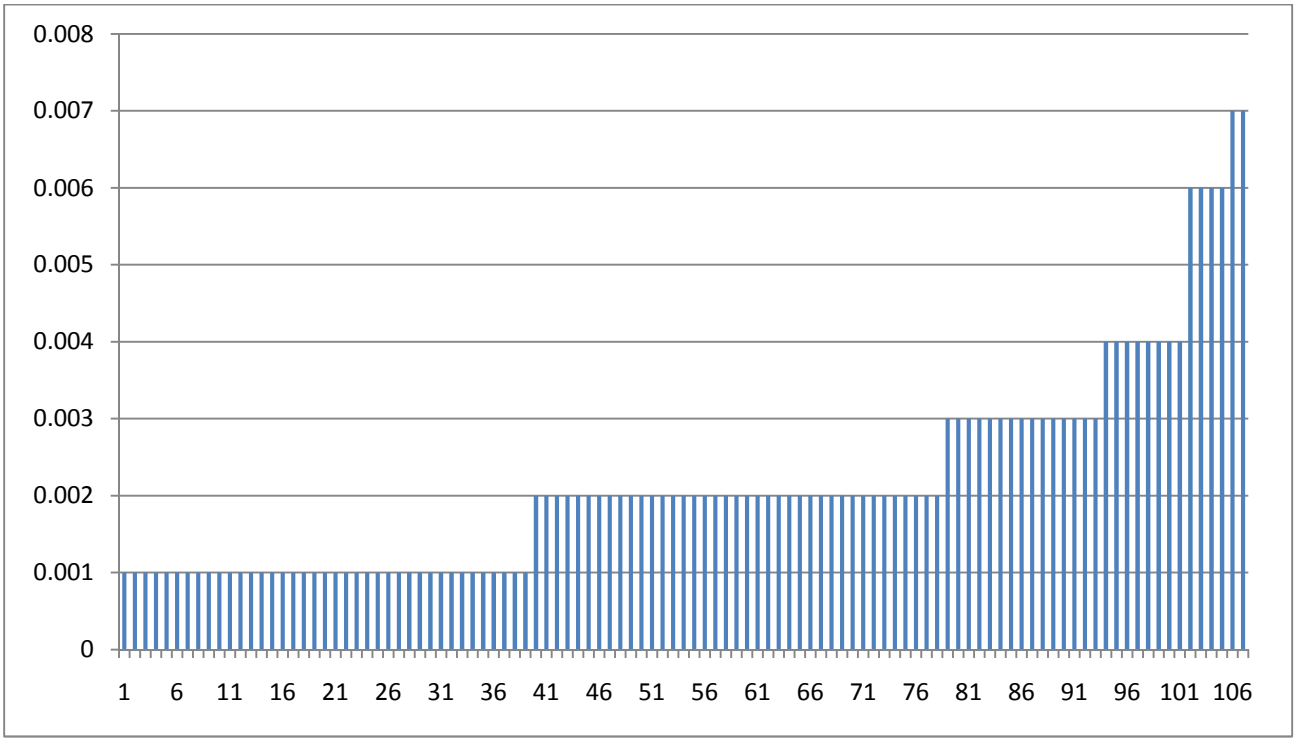

Figure 5.6: State Estimation Accuracy - Load Profile 1 Differences

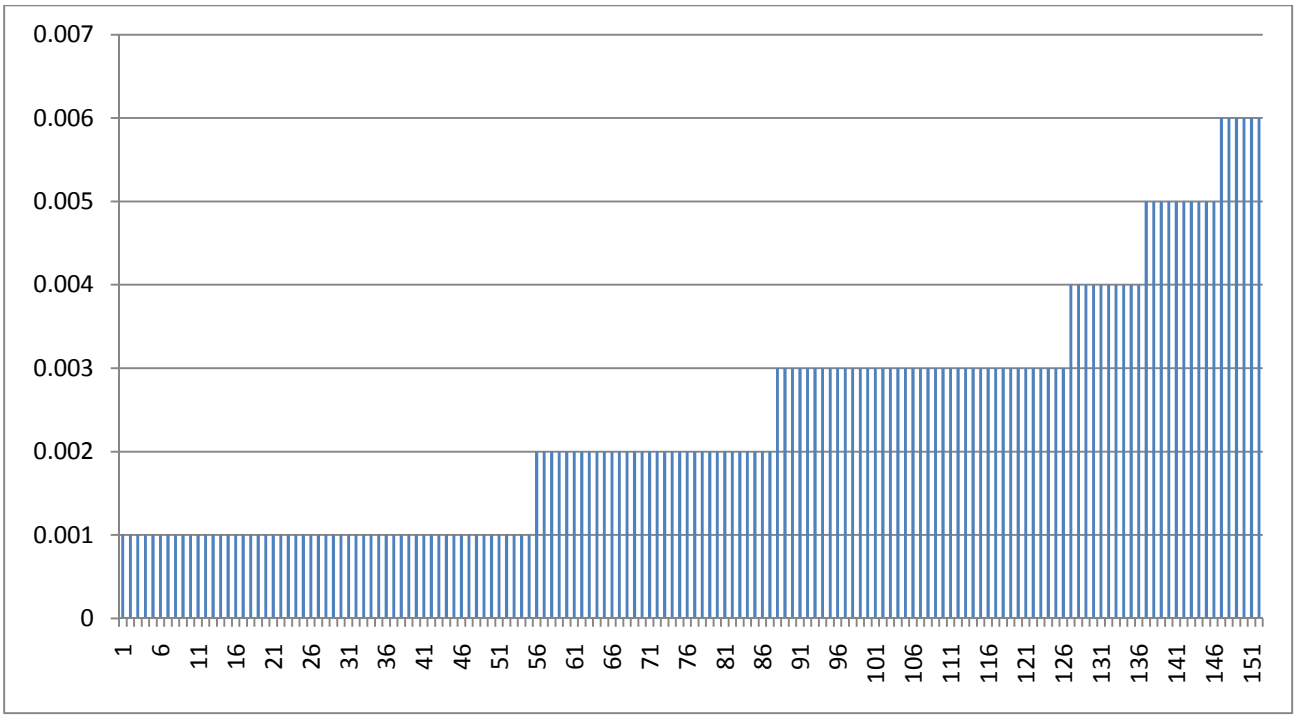

Figure 5.7: State Estimation Accuracy - Load Profile 2 Differences

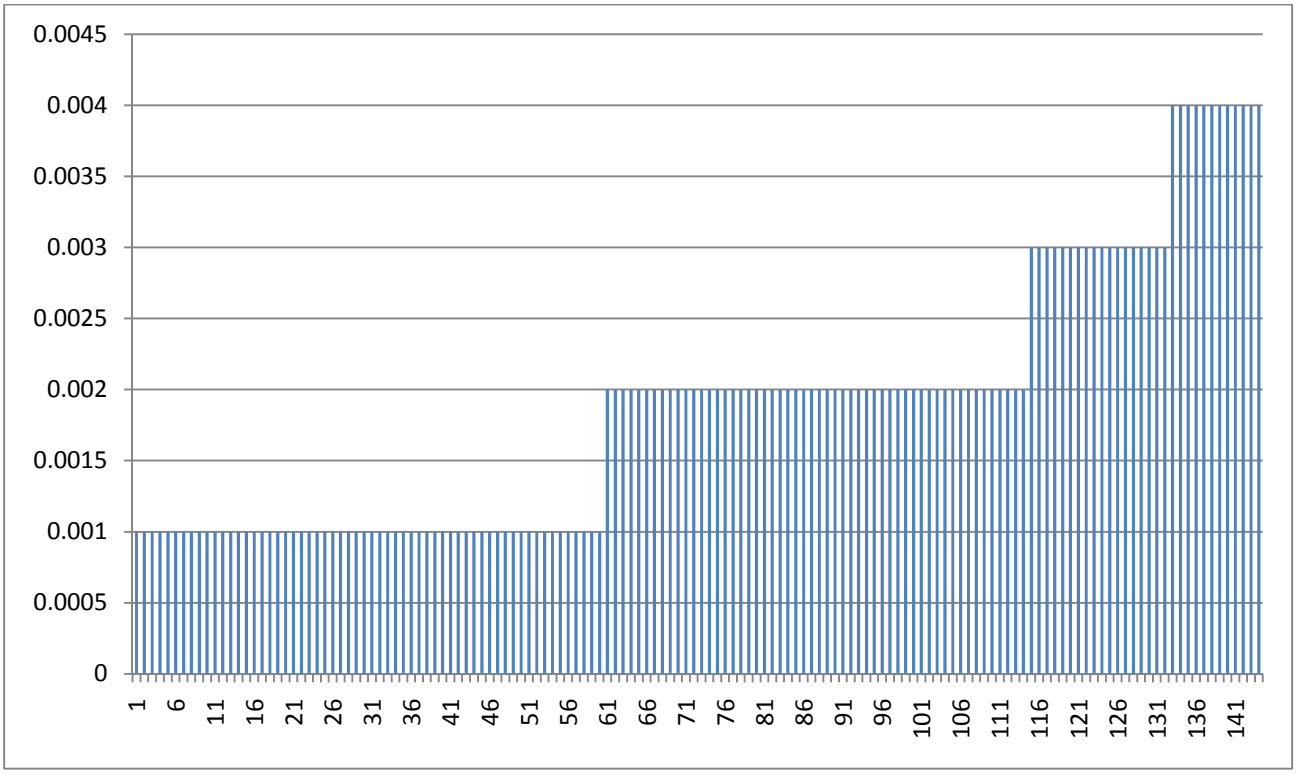

Figure 5.8: State Estimation Accuracy - Load Profile 3 Differences

The results of the accuracy test show that the differences in the final state estimates for each method are almost negligible. In almost all cases, the estimated difference on any given bus is less than the variance of the measurements on that bus. It should be noted that this testing does not determine the absolute accuracy of either method; it only confirms that both methods result in the same state estimate, which shows that reconfigurable monitoring does not result in a loss of accuracy relative to a traditional estimator.

5.4 Fault Tolerance in State Estimation

When faults occur in a power system, the topology data may not be updated to reflect this, or measurements may become unavailable. In such cases, a traditional full-system state estimator may fail to converge on a solution, as not all buses are observed with the remaining measurement set. However, when a system is divided into multiple separate observable areas, then only the estimate for the areas experiencing the fault or measurement problems will have difficulty converging. This allows for a system operator to maintain as much situational awareness as possible while the power system is undergoing a disturbance. The IEEE 300 bus system is used to demonstrate.

Under normal operating conditions, a traditional state estimator converges with an objective value (J) of 2.27 for the injection-only case (real power injection P_i and voltage magnitude V_i at every bus). The computationally islanded system has convergence in all regions. For this test, a breaker failure fault is applied at bus 165, removing it from service. A load flow is performed on the updated system topology to determine the values for the measurement set. The topology for the state estimators, however, is not updated to simulate delay in the trip alarm signals. The resulting objective value of the traditional state estimator is 2.64, indicating that the system reaches a convergence, although the accuracy of the state estimate is decreased. The

computationally islanded system has convergence in all islands except for the one containing the breaker failure fault.

Table 5.5 State Estimation Convergence With a Breaker Failure Fault

Island	*J(x)*	*Convergence*	*Island*	*J(x)*	*Convergence*
1	0.067848	Y	19	0.0533	Y
2	0.016172	Y	20	0.031951	Y
3	0.057128	Y	21	0.024281	Y
4	0.074778	Y	22	0.015799	Y
5	0.084876	Y	23	0.11241	Y
6	0.063625	Y	24	0.076513	Y
7	0.092266	Y	25	N/A	N
8	0.123475	Y	26	0.066189	Y
9	0.073018	Y	27	0.046592	Y
10	0.047403	Y	28	0.073325	Y
11	0.10264	Y	29	0.043074	Y
12	0.021981	Y	30	0.073479	Y
13	0.254118	Y	31	0.086183	Y
14	0.036092	Y	32	0.017766	Y
15	0.077236	Y	33	0.015723	Y
16	0.065463	Y	34	0.015176	Y
17	0.09284	Y	35	0.026988	Y
18	0.0638	Y			

In the divided system, the objective function *J(x)* of each computational will be much smaller than that of the full estimator. The objective function is a summation of error between estimated value and measured value, therefore smaller areas result in smaller *J(x)* values. This variation is a result of the greedy algorithm - some islands are very small, potentially two or three buses. Regardless of island size, the objective values have a larger sensitivity to error than the full system, as can be seen with the convergence failure in the 25th computational island.

Next, multiple faults are applied to simulate a cascade failure. In a real power system, this could happen for many reasons - a large initial disturbance may force under-frequency protection relays to shed load, or outdated relay settings may result in unintentional tripping during a power swing. For this test, buses 47, 86, 97, 114, 165, and 189 are removed and the remaining measurement set is sent to the state estimator. Once again, the topology is not updated, simulating the delay between the event and the system operator's awareness of the event.

Table 5.6 State Estimation Convergence During a Cascade Failure

Island	*J(x)*	*Convergence*	*Island*	*J(x)*	*Convergence*
1	0.071497	Y	19	0.053341	Y
2	0.016171	Y	20	0.031947	Y
3	0.062318	Y	21	0.024278	Y
4	0.094736	Y	22	0.015803	Y
5	0.104263	Y	23	0.11241	Y
6	0.076452	Y	24	N/A	N
7	0.121576	Y	25	N/A	N
8	0.263475	Y	26	0.074233	Y
9	0.084263	Y	27	0.046592	Y
10	0.047401	Y	28	0.082457	Y
11	0.102629	Y	29	0.057031	Y
12	0.021975	Y	30	N/A	N
13	0.254115	Y	31	0.086183	Y
14	0.036094	Y	32	0.017765	Y
15	0.077235	Y	33	0.015725	Y
16	0.065461	Y	34	0.015176	Y
17	0.092837	Y	35	0.026988	Y
18	0.063955	Y			

The traditional state estimator fails to converge - too many state variables are now undetermined and cannot be reconciled. The estimation performed with reconfigurable monitoring has 3 islands which fail to converge, but the other 32 are still observable, although the objective values are higher than in the case of the breaker failure fault. This knowledge may

assist the system operator in reestablishing the stability of the power system. This result does not imply that a traditional state estimator cannot be designed to compensate for such events, only that it does not inherently do so.

Chapter 6. Conclusion

6.1 Summary

The concept of reconfigurable monitoring - the ability to divide a power system into separate areas, called computational islands, and analyze different groups as necessary - has been presented in this thesis. Following background discussion on state estimation and phasor measurement units, the necessary computations involved in a traditional state estimator were explained. System equivalencing was shown to divide a system into multiple parts, which allow for a state estimator to operate in regions rather than on an entire system. Expanding upon this technique, the utilization of a PMU as a computational border was explained. Using a PMU in this manner requires that instrument transformers be properly calibrated and that a method of bad data checking exists to ensure that the synchrophasors remain accurate. Techniques for accomplishing this were presented and mathematically verified.

To test the effectiveness of reconfigurable monitoring, a PMU placement algorithm was developed to determine the placement and installation order of PMUs to maximize the number of computational islands while working with existing PMUs. To ensure that only necessary buses were considered in the placement algorithm, a set of rules for system reduction was presented. These rules remove extraneous buses which are not valid locations for PMU installation or necessary to determine system state. The PMU placement algorithm was executed on 3 different test systems, creating placement sets for the IEEE 118 bus system, IEEE 300 bus system, and the reduced (1457 bus) Brazilian ONS transmission system. These systems were used to verify the operation of reconfigurable monitoring. Islanding aggregation and reconfiguration was presented using the IEEE 118 bus system. State estimation speed and accuracy tests were performed with the ONS system, and state estimation fault tolerance testing was performed with the IEEE 300

bus system. In the state estimation tests, the results were compared with the performance of a traditional state estimation algorithm. The testing shows that state estimation speed can be dramatically improved, relative to a traditional state estimator, and that the solution for the state variables is the same. Additionally, the fault tolerance test shows that reconfigurable monitoring allows for observation of the majority of the power system as a disturbance is occurring.

6.2 Future Work

While the research presented here is intended to provide a comprehensive overview of computational isolation/islanding, and reconfigurable monitoring, it is by no means complete. The algorithm discussed in Chapter 4 is only meant to maximize computational islanding, but may not be entirely applicable in a real power system, as system planners may have additional considerations involved for PMU placement. Additionally, the applications provided in Chapter 5 are only examples, there are potentially many more ways to integrate reconfigurable monitoring into other applications. The following recommendations are made to expand upon the research presented in this thesis:

- Further testing and validation of remote instrument transformer calibration methods.
- Exploration of further uses for the flexibility afforded by reconfigurable monitoring; adaptive protection and online pre-emptive contingency analysis are two functions which could possibly be improved.
- Development of a comprehensive multi-function PMU placement algorithm, one which will determine placement according to a more robust set of criteria. Computational isolation, observability, voltage stability, and economic viability should all be taken into account. Such an algorithm would likely involve heuristic and combinatoric methods, and not rely on a simple "greedy" process.

- Development of a parallel state estimation algorithm for use with reconfigurable monitoring. With sufficient hardware, the computational islands could all be processed at once, rather than serially. New parallel computation techniques utilizing Graphical Processor Units (GPUs) could prove useful here. If state estimation runtime can be reduced enough, a system-wide dynamic state estimator may be feasible.

References

[1] A.G. Phadke, *Synchronized Phasor Measurements in Power Systems*, Computer Applications in Power, IEEE, vol. 6, pp. 10-15, 1993.

[2] Reynaldo Francisco Nuqui, "State estimation and voltage security monitoring using synchronized phasor measurements," Ph.D. dissertation, Virginia Polytechnic Institute & State University, Blacksburg, VA, USA, 2001.

[3] "IEEE Standard for Synchrophasors for Power Systems," IEEE Std 1344-1995, 1995.

[4] "IEEE Standard for Synchrophasors for Power Systems," IEEE Std C37.118-2005, 2005.

[5] L. Mili, T. Baldwin, and R. Adapa, "Phasor Measurement Placement for Voltage Stability Analysis of Power Systems, in Decision and Control, 1990., Proceedings of the 29th IEEE conference on, 1990, pp. 3033-3038 vol.6.

[6] A. Abur and A. G. Exposito, *Power System State Estimation Theory and Implementation*, CRC Press, 2004.

[7] D. Kirschen and G. Strbac, "Why Investments do not Prevent Blackouts," http://www.ksg.harvard.edu/hepg/Standard_Mkt_dsgn/Blackout_Kirschen_Strbac_082703.pdf

[8] F. C. Schweppe, "Power System Static-State Estimation, Part III: Implementation," IEEE Transactions on Power Apparatus and Systems, vol. PAS-89, pp. 130-135, 1970.

[9] D. P. Kothari, I. J. Nagrath, *Modern Power System Analysis*, 3rd ed., Tata McGraw-Hill, New Delhi, 2003.

[10] A. G. Phadke, J.S. Thorp, "History and Application of Phasor Measurements," Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES, 2006, pp 331-335.

[11] A. G. Phadke, J.S. Thorp, *Synchronized Phasor Measurements and Their Applications*, New York: Springer-Verlag, 2008.

[12] Ming Zhou, "Advanced system monitoring with phasor measurements," Ph.D. dissertation, Virginia Polytechnic Institute & State University, Blacksburg, VA, USA, 2008.

[13] "IEEE Standard Requirements for Instrument Transformers," IEEE Std C57.13-2008, pp. 14-15, 2008.

[14] J.H. Harlow, *Electric Power Transformer Engineering*, CRC Press, 2004.

[15] J. Chen, A. Abur, "Placement of PMUs to Enable Bad Data Detection in State Estimation," *Power Systems, IEEE Transations on,* Vol. 21, No. 4. (2006), pp. 1608-1615.

[16] Siemens PTI, *Power System Software for Engineering*, 1976-2009.

[17] James Ross Altman, "A practical comprehensive approach to PMU placement for full observability" M.S. thesis, Virginia Polytechnic & State University, Blacksburg, VA, USA, 2007.

Appendix A. State Estimation Equations

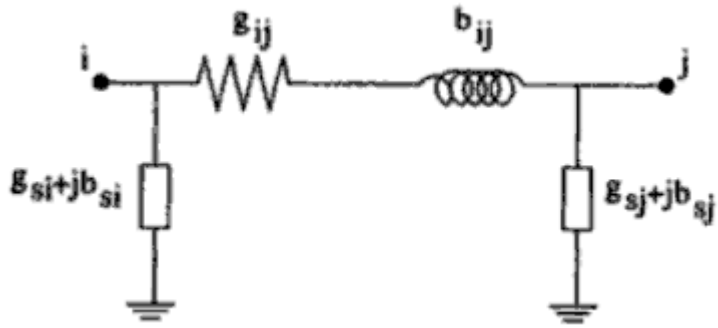

Figure A.1: Network Branch π Model

Given a two-port π-model of a network branch, as shown above, the equations used in a traditional state estimator are as follows:

Measurement Function Equations

$$P_i = V_i \sum_{j=1}^{N \neq i} V_j \left(G_{ij} cos\ \theta_{ij} + B_{ij} sin\ \theta_{ij}\right) \tag{A.1}$$

$$Q_i = V_i \sum_{j=1}^{N \neq i} V_j \left(G_{ij} sin\ \theta_{ij} - B_{ij} cos\ \theta_{ij}\right) \tag{A.2}$$

$$P_{ij} = V_i^2 \left(G_{si} + G_{ij}\right) - V_i V_j \left(G_{ij} cos\ \theta_{ij} + B_{ij} sin\ \theta_{ij}\right) \tag{A.3}$$

$$Q_{ij} = -V_i^2 \left(B_{si} + B_{ij}\right) - V_i V_j \left(G_{ij} sin\ \theta_{ij} - B_{ij} cos\ \theta_{ij}\right) \tag{A.4}$$

Jacobian Matrix (H) Equations

$$\frac{\partial P_i}{\partial \theta_i} = \sum_{j=1}^{N} V_i V_j \left(-G_{ij} sin\ \theta_{ij} + B_{ij} cos\ \theta_{ij}\right) - V_i^2 B_{ii} \tag{A.5}$$

$$\frac{\partial P_i}{\partial \theta_j} = V_i V_j \left(G_{ij} sin\ \theta_{ij} - B_{ij} cos\ \theta_{ij}\right) \tag{A.6}$$

$$\frac{\partial P_i}{\partial V_i} = \sum_{j=1}^{N} V_j \left(G_{ij} cos\ \theta_{ij} + B_{ij} sin\ \theta_{ij}\right) + V_i G_{ii} \tag{A.7}$$

$$\frac{\partial P_i}{\partial V_j} = V_i\left(G_{ij}cos\ \theta_{ij} + B_{ij}sin\ \theta_{ij}\right) \tag{A.8}$$

$$\frac{\partial Q_i}{\partial \theta_i} = \sum_{j=1}^{N} V_i V_j\ \left(G_{ij}cos\ \theta_{ij} + B_{ij}sin\ \theta_{ij}\right) - V_i^2 G_{ii} \tag{A.9}$$

$$\frac{\partial Q_i}{\partial \theta_j} = V_i V_j\left(-G_{ij}cos\ \theta_{ij} - B_{ij}sin\ \theta_{ij}\right) \tag{A.10}$$

$$\frac{\partial Q_i}{\partial V_i} = \sum_{j=1}^{N} V_j\ \left(G_{ij}sin\ \theta_{ij} - B_{ij}cos\ \theta_{ij}\right) - V_i B_{ii} \tag{A.11}$$

$$\frac{\partial Q_i}{\partial V_j} = V_i\left(G_{ij}sin\ \theta_{ij} - B_{ij}cos\ \theta_{ij}\right) \tag{A.12}$$

Appendix B. PMU Placement Algorithm Code

PMUPlace.m:

```
% Program to optimally place PMUs with regards to computational islanding

%loop through all buses
%check for placed PMU
%place temp PMU, count islands

%save max number of islands created and bus

PMULimit = 125;
NewPMULimit = PMULimit - PMUCounter;
Calibrate = 0;
VoltageCheck = 0;

for Ctr = 1:NewPMULimit,

    tic

    MaxIslands = 0;
    MaxIslandBus = 0;

    for i0 = 1:length(Bus.Number),
        if(Bus.HasPMU(i0) == 0 && Bus.IsTerminal(i0) == 0 ...
                && Bus.Ignore(i0) == 0),    % if the bus does not already
have a PMU and is being considered
            Bus.HasPMU(i0) = 1;         % place a temporary PMU
            IslandCount;                % and count the islands
            Bus.NumIslands(i0,1) = NumIslands;

            if(NumIslands > MaxIslands),
                MaxIslands = NumIslands;
                MaxIslandBus = i0;
            elseif(VoltageCheck == 1 && NumIslands == MaxIslands),
                if(Bus.BasekV(i0,1) > Bus.BasekV(MaxIslandBus,1)),
                    MaxIslands = NumIslands;
                    MaxIslandBus = i0;
                end
            end

            %if(NumIslands ~= 0),
            %    fprintf('PMU at Bus %d\n',i0);
            %    NumIslands
            %end

            Bus.HasPMU(i0) = 0;         % then remove the temporary PMU
            for(i1 = 1:length(Bus.Number)),
                Bus.Visited(i1) = 0;
            end
        else
            % nothing required
        end
```

```
    end

    Bus.HasPMU(MaxIslandBus) = 1;
    fprintf('Iteration %d Result: PMU placed on bus %d\n',Ctr,MaxIslandBus);
    fprintf('Islands created: %d\n',MaxIslands);

    if(Calibrate == 1),
        CalibratePoint = median(Bus.NumIslands) + 1;
        for i0 = 1:length(Bus.Number),
            if(Bus.NumIslands(i0,1) > CalibratePoint),
                Bus.Ignore(i0) = 0;
            else
                Bus.Ignore(i0) = 1;
            end
        end
        Calibrate = 0;
    end

    toc
end
```

IslandCount.m:

```
%Counts the total number of computational islands in the system for a given
%PMU layout

NumIslands = 0;

for i1 = 1:length(Bus.Number);
    if(Bus.HasPMU(i1) == 1),
        NumIslands = NumIslands + DFS_Start(i1);
    else
        % no else
    end
end
```

DFS_Start.m:

```
function [ numIslands ] = DFS_Start( TargetBus )
%Depth First Search to traverse a power system, starting at a PMU
global Bus Branch BAM;

numIslands = 0;

for i0 = 1:Bus.NumBranches(TargetBus),
    if(Bus.Visited(BAM(TargetBus,i0)) == 0 && Bus.HasPMU(BAM(TargetBus,i0))
== 0),
        DFS(BAM(TargetBus,i0));
        numIslands = numIslands + 1;
    else
        % no else required
    end
```

```
end
```

DFS.m:

```
function [ IsIsland ] = DFS( TargetBus )
%Depth First Search to traverse a power system
global Bus Branch BAM;

if(Bus.Visited(TargetBus) == 1),
    return;
elseif(Bus.HasPMU(TargetBus) == 1),
    return;
else
    Bus.Visited(TargetBus) = 1;
    for i0 = 1:Bus.NumBranches(TargetBus),
        DFS(BAM(TargetBus,i0));
    end
    %TargetBus
end
```