Automated Topology Processing for Conventional, Phasor-Assisted and Phasor-Only State Estimators

February, 2012

MOSTAFA FARROKHABADI

KTH Electrical Engineering

Master's Degree Project
Stockholm, Sweden 2012

XR-EE-ES
2012:003

August 2011 to February 2012

Automated Topology Processing for Conventional, Phasor-Assisted and Phasor-Only State Estimators

— Diploma Thesis —

Mostafa Farrokhabadi

Electrical Power Systems Division

School of Electrical Engineering, KTH Royal Institute of Technology, Sweden

Supervisor
Dr. Luigi Vanfretti
KTH Stockholm

Examiner
Dr. Luigi Vanfretti
KTH Stockholm

Stockholm, February, 2011

Abstract

Although the "State Estimation" concept was established decades ago, it is still a cornerstone for developing advanced applications for real-time operation and control of transmission networks. State Estimation is a key Energy Management System (EMS) function responsible of providing static estimates of the system states and line active and reactive power flows. Traditionally, state estimation uses asynchronous measurements of active and reactive power flows and voltage magnitudes, using the ICCP protocol. However, there are some novel ideas emerging recently, which suggest the usage of Phasor Measurment Unit (PMUs) that provide synchronous Measurements of positive sequence voltage and current phasors at a sample rate of up to 120 per second. The application of PMUs in power systems proliferated after the major blackouts in 2003, which revealed the critical need for wide area monitoring systems which gather data synchronously from several Phasor Measurement Units, and to give a wide area awareness of large power networks.

One of the key steps at each state estimation procedure takes place is Topology Processing. Topology processing refers to the determination of the system topology through the available data from the status of the network through bus breaker status, etc. and its link to a static database which holds relevant information about the parameters of such topology. It is one the most critical steps before any other analysis can be carried out at an energy management system (EMS). At the first glance, it may seem quite easy to determine the system's topology: an open breaker shows that a line is disconnected. However, the problem is much more challenging when it comes to reality. Considering a *real* power system, there are numerous substations with different configurations, and all of those stations are connected to each other. On top of that, there may be instances when an interconnected power system is split in separate islands. The task of a network topology processor is to correctly detect these changes, and create an equivalent network model. The output of a topology processor then can be used by an EMS application, such as a state estimator. Network analysis software need to interpret the output of the topology processor and solve the associated equations related to current topology of the network which was provided by the topology processor. To make this interpretation possible, network topology processors need to transform the bus-section/switching-device model of the network into the bus/branch model. This bus/branch model then can be used for further network analysis.

The work reported in this thesis encompasses development of a robust network topology processor which can be used for both traditional and PMU-based state estimators. Firstly, previous research in the field of topology processing is critically scrutinized, and the drawbacks are identified and discussed. Building on top of the state of the art an algorithm is proposed in a way to cover the limitations of current approaches and to suggest new features. The algorithm is robust enough to work with traditional data with or without PMU measurements, or to operate solely using PMU data. The topology processor was coded in MATLAB and tested over two different power networks including IEEE Reliability Test System 1996. As the topology processor is designed to supply network topologies to a PMU-based Sate Estimator, the IEEE Reliability Test System 1996 is simulated in real-time, as a bilateral work for future implementation of a complete PMU-based state estimator. Different test scenarios are performed and the voltage and current phasor outputs are extracted to emulate PMU measurements. Real-time simulation is performed using Opal-RT real-time simulator which is part of the "SmarTS Lab" at KTH Royal Institute of Technology.

Chapter 1: Introduction

1.1. Background

In the year 2000 the National Academy of Science named "Electrification" as the single "Greatest Engineering Achievement of the 20th Century", with no doubt, power systems are the most complex systems ever made by humans. At the same time that the demand for energy supply continues to increase all over the world, power networks are also transforming into much more intricate systems comprised of large interconnected areas. In addition, the increase of energy prices, shortage in energy resources and environmental constraints, put unparalleled pressures for the economical, reliable and secure operation and control of transmission grids. Recently, the concept of "Smart Grids", that is electricity networks compromised of integrated communication systems, advanced sensing, measurement infrastructure, decision support and human interfaces [29], promises the availability of new mechanisms to counter-act the increasing pressures that the current electricity systems are subject to, as explained above.

Although the "State Estimation" concept was established decades ago [2], it is still a cornerstone for developing advanced applications for real-time operation and control of transmission networks. State Estimation is a key Energy Management System (EMS) function responsible of providing static estimates of the system states, i.e., bus voltages and angles, and line active and reactive power flows [3]. State estimation is a term used in power engineering not only as a solution of the system states alone, but also it has comprises other functions such as topology processing, observability analysis, and bad data processing. Traditionally, State Estimation uses asynchronous measurements of active and reactive power flows and voltage magnitudes[1]. It gathers data using the TCP/IP protocol, through communication networks using a star-round-robin polling mechanism that centralizes data, as such, an asynchronous data set is captured at sample rate typically of 1 to 10 seconds. However, there are some novel ideas emerging recently, which suggest the usage of Phasor Measurment Units to provide synchronous data at sample rate of up to 120 per second.

One of the key steps at each state estimation procedure takes place is Topology Processing. Topology processing refers to the determination of the system topology through the available data from the status of the network through bus breaker status, etc.

[1] In some limited cases the inclusion of line current magnitude measurements it is also carried out, however, this practice is less common than expected.

and its link to a static database which holds relevant information about the parameters of such topology. It is one the most critical steps before any other analysis can be carried out at an energy management system (EMS). At the first glance, it may seem quite easy to determine the system's topology: an open breaker shows that a line is disconnected. However, the problem is much more challenging when it comes to reality. Considering a *real* power system, there are numerous substations with different configurations, and all of those stations are connected to each other. On top of that, there may be instances when an interconnected power system is split in separate islands. The task of a network topology processor is to correctly detect these changes, and create an equivalent network model. The output of a topology processor then can be used by an EMS application, such as a state estimator. Network analysis software need to interpret the output of the topology processor and solve the associated equations related to current topology of the network which was provided by the topology processor. To make this interpretation possible, network topology processors need to transform the bus-section/switching-device model of the network into the bus/branch model. This bus/branch model then can be used for further network analysis

Since the introduction of PMUs, there have been lots of proposals for using phasor measurement data for state estimation [3]. Two approaches have been introduced, both of them relying only PMU data to perform state estimation [4, 5]. In comparison with conventional state estimation, these approaches use Kirchoff laws instead of power flow models. Additionally, they perform state estimation with much higher rates compare to traditional ones, i.e. 30-60 times per second [3]. These approaches are known as linear state estimation [4], and Phasor State Estimation (PSE) [5] and the solution algorithm is different for each one. Linear state estimation, as its name suggests, uses a weighted linear least squares solution, while Phasor State Estimation apply a nonlinear WLS estimator.

The above mentioned researches increase the need for a topology processor that is robust enough to not only work with TCP/IP data but also has the ability of working with PMU data whenever needed. In the coming section, there is a brief review over the main published researches both in the area of state estimation and topology processing.

1.2. Literature Review

After Schweppe and Handschin firstly introduce the concept of State Estimation during 1970's [1], many other studies have been published to enhance its performance. This includes many diverse efforts such as works on observability [18][19][20], or bad data processing [21][22][23]. Specifically, two complete books are written about state estimation which includes the most relevant topics in the field [2][6].

Since the introduction of PMU, many methods have been proposed for using PMU measurements in state estimation algorithm [7]. While some algorithms have provided a hybrid solution in which both ICCP and PMU measurements have been used, there are two major PMU-only based methods [4][5]. In [4] the proposed method is based on simple linear Kirchhoff's law of voltage and current which is solved by a linear least

square solution. In [5] the non-linear weighted least square solution is used by application of iterative algorithms such as Guass-Newton; here the main difference is that complex power equations are replaced by Kirchhoff's laws. The main advantage of the latter paper over the former is its ability to detect and fix angle bias errors in PMU measurements.

The main issue with most of the works on state estimation is that they have not properly addressed issues related to topology processing, which is one of the most critical steps before being able to perform state estimation. In addition, the number of available studies specifically about network topology processing are very limited and most of them do not offer the necessary components for a practical implementation. One of the very first papers regarding topology processing was published by Sasson in 1973 which is still the reference work for many other proposals [8]. The proposed algorithm in [8] is based on matrices as measurement and connectivity as input. In [9] all the necessary steps involved in real time modeling of power systems are descriptively explained. This includes some very general explanations regarding network topology processing. In [10] an algorithm is proposed which is the same as one proposed in [8], with the exception that it saves the output of the topology processor from one cycle to the next and performs some comparisons to determine if any major change has occurred in the network or not. In the case of small changes, there is no need to refactorize all of the matrices in the network analysis software which uses the output of the network topology processor. In [11] a new algorithm is proposed which uses artificial intelligence and neuronal networks to determine the network topology. This is a very complex method which seems to be far from being practical. A graph based algorithm is suggested in [12] which just uses PMU data in order to determine the network's topology. The algorithm proposes a new approach; however it lacks the ability to deal with changes inside the substation efficiently. Morever, as the algorithm is using the current threshold through the lines to check for connectivity, it can't be used when just TCP/IP data is available. In [13] a generalized approach for state estimation is defined which involves all switch and breaker measurements available. When it comes to the topology processor part, just a very brief explanation is provided and no algorithm or practical procedure is described. Also in [16], another topology engine is proposed which is based on graph theory. It transforms graph data to a sparse matrix by an algorithm and uses sparse matrix theories to solve for the topology of the network. This is probably a fast method to determine the network topology. However, the approach lacks the ability to deal with the changes inside a substation. In addition, the proposed algorithm is far more complex than the other matrix based algorithms, which makes the explanation of the method quite vague; this results in difficulties for its implementation. Also in [6] there is a description on a topology processor. However, the presentation is just descriptive explanations of the topology processor and a definition of the general procedure and outputs.

The eMegaSim OPAL-RT real-time simulator is used for real time simulation in this thesis. The documentation is provided in [27]. SimPowerSystems, the proprietary software used to simulate power network in this study, available in MATLAB/Simulink contains a library for modeling and simulating an electric power network. The relative information regarding SimPowerSystems can be found [67].

1.3. Objectives

1.3.1. General Objectives

The main objective of this study is to develop an algorithm that is capable of determining the network topology using breakers status and network connectivity data. This algorithm should have the ability of working with TCP/IP, a combination of TCP/IP and PMU, or PMU-only measurements. The algorithm has been developed fir its use in state estimation, both conventional and PMU-only.

The algorithm should perfectly deal with changes either inside or outside of a substation. It should detect whenever a substation is split to new stations, or when some substations merge together and form a new one. In addition, it should automatically assign numbers to new substations. Also, the method should be capable of detecting if an interconnected network is split to several separated islands, and whether each island is energized or not. To add to this, the islands also should be specified numbers automatically.

The proposed topology processor should be tested with different test power networks, while performing several different breaking switching scenarios. The results should be correct and satisfy all the pre-mentioned constraints.

Finally, the IEEE reliability test system 1996 [14], should be implemented and simulated in real-time. This is due to the aim of the thesis that the results are going to be used for an implementation of a complete package for PMU-only state estimation. Enough test scenarios should be made, and measurements containing the states of the system should be saved to act as PMU data for possible future applications.

1.3.2. Specific Objectives

- To perform a comprehensive literature review and identify the drawbacks with the previous proposed methods for topology processing.
- To implement, in functioning software, a topology processor algorithm that addresses the limitation of current TPs.
- The proposed new algorithm should be able to work both with TCP/IP, TCP/IP and PMU, or only PMU measurements.
- Specifically, the algorithm should be tested with two different test systems. One is IEEE Reliability Tests System 1996 [14], and the other is the same as that proposed in [8].
- The IEEE Reliability Test system should be implemented in real time, and tested with different scenarios to show the efficiency of the measurements to act like

PMU measurements. This simulation is going to be used for future implementation of PMU based state estimation.

1.4. Outline

All the chapters in this thesis start with a brief introduction of the contents that presented within that chapter. Just after the introduction, a table of figures comes containing information regarding the number and title of the figures used in that chapter. Each chapter ends with a brief summary; this part contains a concise explanation over what was discussed in the chapter and possible conclusions.

Chapter 2 contains information regarding state estimation itself. It fully explains the procedure and algorithms used in both conventional and PMU-only state estimation. The weighted least square and maximum likelihood algorithms are described along with the modeling of the state and measurement equations. However, no information is provided regarding bilateral functions of a state estimator such as topology processor or network observability. Chapter 3 is related to the network topology processor. The concept of network topology is fully explained. Different aspects of a network topology processor such as processing of input data, substation analysis, islanding analysis, and energization analysis are completely described. Chapter 4 presents information regarding power system modeling and implementation in real-time. It fully describes the procedure of working with the Opal-RT real time simulator, and the implementation of the test system in Simulink and in real-time subsequently. The steps of modeling in SimPowerSystems are also demonstrated. Chapter 5 in devoted to the exposition of a proposed algorithm for topology processing. First, the drawbacks of current available topology processor are identified. The new algorithm is explained step by step. Results of the application of the new algorithm to the tests systems are provided in Chapter 6. The parameters and characteristics of the systems along the waveforms from real time simulation are also provided. Finally, a conclusion for the whole thesis and suggestions for possible future works are discussed in chapter 7.

Chapter 2: Power System State Estimation

2.1 Introduction

This chapter provides information regarding the state estimation in power systems. First, the chapter starts with explanation regarding the conventional state estimation. Problem formulation, maximum likelihood estimation and measurement equation modeling all have been included.

Next, PMU-only state estimators have been discussed. Two different proposed algorithms on the field including Linear and Phasor State Estimation have been discussed. The chapter ends with a brief summary over its contents.

2.2. State Estimation

Although the "State Estimation" concept was proposed decades ago [2], it is still a cornerstone for developing advanced applications for real-time operation and control of transmission networks. State Estimation is a key Energy Management System (EMS) function responsible of providing static estimates of the system states. One of the key goals behind any state estimator is to compensate for any inaccuracy in the measurements, and provide reliable system states than can be further used for contingency analysis, optimal power flow, etc. Additionally, each state estimator is comprised of other functions which are vital for proper operation of the estimator. Fig 2-1 is a simple representation of a complete state estimator.

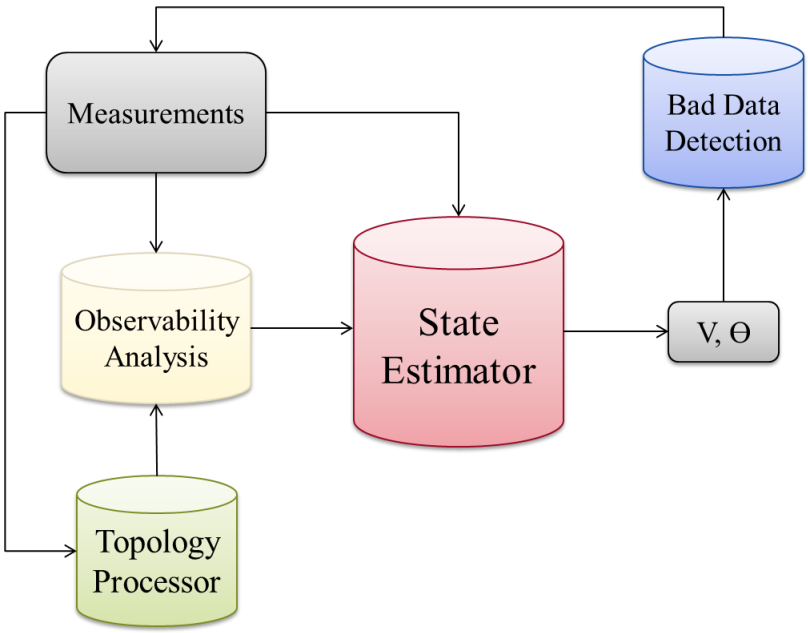


Fig. 2-1: State Estimator Functions

As it can be seen in the figure above, a typical state estimator includes:

- Topology processor: determine the topology of a network using the breakers status and connectivity data.
- Observability analysis: using the set of available measurements, it determines if a state estimation solution can be found for whole the network; otherwise, it determines the islands or branches than can't be observed.
- Bad data detection: Detect any major errors in the measurements, and eliminate measurements with an unacceptable accuracy.

Normally, a state estimator is in charge of estimating the following states [6]:

- Voltage magnitude and angle at every substation
- Turn ratio magnitude and angle at every transformer

Traditionally, State estimation uses asynchronous measurements of active and reactive power flows and voltage magnitudes[2]. It gathers data using the ICCP protocol, through communication networks using a star-round-robin polling mechanism that centralizes data, as such, an asynchronous data set is captured at sample rate typically of 1 to 10 seconds. The measurements that are used in conventional state estimators are usually voltages magnitude, complex powers either as an injection or branch flow, and magnitude of currents through the branches. The problem formulation and how to solve the estimation problem are fully explained below.

*2.2.1. **Problem Formulation***

The conventional state estimator considers the following nonlinear equation:

$$z = h(x) + e \tag{2.1}$$

Here, *z* is the vector of measurements consisting of *m* different measurements, *x* is the state vector, and *h* is the vector of nonlinear equations relating the state variables to measurements. *e* is also the vector of measurement errors. There are two assumptions about the elements of the measurement error vector [2]:

- $E(e_i) = 0, \quad i = 1,...,m.$
- $E[e_i e_j] = 0 \Rightarrow Cov(e) = E[e.e^T] = R = diag\left\{\sigma_1^2, \sigma_2^2, ..., \sigma_m^2\right\}.$

The object of state estimator is to minimize the following objective function [2]:

$$J(x) = \sum_{j=1}^{m}\left(\frac{z_j - h_j(x)}{\sigma_j}\right)^2 = \left[z - h(x)\right]^T R^{-1}\left[z - h(x)\right] \tag{2.2}$$

To find the minimum of the function, the first order derivative of the function should be equal to zero [2]:

$$g(x) = \frac{\partial J(x)}{\partial x} = -\left[\frac{\partial h(x)}{\partial x}\right]^T R^{-1}\left[z - h(x)\right] = 0 \tag{2.3}$$

By expanding the nonlinear function g(x) into its Taylor series around the state vector x^k and neglecting the higher orders, the following equation is derived [2]:

$$x^{k+1} = x^k - \left[G(x^k)\right]^{-1}.g(x^k) \tag{2.4}$$

[2] In some limited cases the inclusion of line current magnitude measurements it is also carried out, however, this practice is less common than expected.

Where:

$$G(x^k) = \frac{\partial g(x^k)}{\partial x} = H^T(x^k) \cdot R^{-1} \cdot H(x^k)$$
$$g(x^k) = -H^T(x^k) \cdot R^{-1} \cdot \left(z - h(x^k)\right)$$
$$H(x) = \left[\frac{\partial h(x)}{\partial x}\right]$$

Here G(x) is called the "gain matrix" [2]. Replacing $g(x^k)$ into the equation (2.4) leads to the following equations set which are called as "Normal Equations" [2]:

$$\left[G(x^k)\right]\Delta x^{k+1} = H^T(x^k)R^{-1}\left[z - h(x^k)\right] \qquad (2.5)$$

The Gauss-Newton iterative solution can be used to solve the set of Normal Equations (2.5).

2.2.2. WLS Solution

The set of Normal Equations introduced in the previous section can be solved iteratively using Weighted Least Square method. The algorithm can be summarized as follows [2]:

1- Set the state vector x^k as flat, i.e. all voltages magnitude are assumed to be equal to 1 PU and their corresponding angle is set to 0.
2- Calculate the gain matrix.
3- Calculate the right hand side of equation (2.5).
4- Find Δx^k.
5- Investigate if Δx^k is less than a predefined convergence limit. If yes, the algorithm is finished; otherwise go to step 6.
6- Set $x^{k+1} = x^k + \Delta x^k$, and go back to step 3.

Notice:

To find for Δx^k in step 4, the inverse of the gain matrix should be calculated; however, as the gain matrix is usually a sparse matrix, there are easier methods to calculate its inverse such as Cholesky Decomposition method [30].

2.2.3 Measurements equations

As mentioned previously, in a conventional state estimation measurements are limited to the voltage and current magnitudes and complex power injection or flows through the lines. As PMUs emerged, it is now also possible to measure the voltage and current phasors, i.e. both magnitudes and angles. However, the application of PMUs in state estimation is fully discussed in section 2.3. So the measurements considered here are only those used in conventional SE.

2.2.3.1 The measurement function matrix, h(x)

Usually by the state of system we refer to bus voltages and angles. The measurement function consists of those equations which relate the states of the system to the measurements. In most cases measurements consist of voltage magnitude, current magnitude through the lines, line complex power flow and complex power injected at different buses. The equations regarding each type of the mentioned measurements are provided in the following expressions:

- Complex power injections at bus *i*:

$$\begin{aligned} P_i &= V_i \sum_j V_j \left(G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij} \right) \\ Q_i &= V_i \sum_j V_j \left(G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij} \right) \end{aligned} \tag{2.6}$$

Notice:

In equation (2.6), Buses *j* are all those ones which are connected to bus *i*.

- Complex power flow from bus *i* to bus *j*:

$$\begin{aligned} P_{ij} &= V_i^2 \left(g_{sh-i} + g_{ij} \right) - V_i V_j \left(g_{ij} \cos\theta_{ij} + b_{ij} \sin\theta_{ij} \right) \\ Q_{ij} &= -V_i^2 \left(b_{sh-i} + b_{ij} \right) - V_i V_j \left(g_{ij} \sin\theta_{ij} - b_{ij} \cos\theta_{ij} \right) \end{aligned} \tag{2.7}$$

- Current magnitude from bus *i* to bus *j*:

$$I_{ij} = \frac{\sqrt{P_{ij}^2 + Q_{ij}^2}}{V_i} \tag{2.8}$$

In equations (2.6) to (2.8), V and Θ represent the voltage magnitude and angle at the corresponding bus; G and B are the real and imaginary elements of the complex bus admittance matrix respectively; g and b are the real and imaginary part of the line admittance connecting bus i to bus j; finally, g_{sh} and b_{sh} are representing the admittance of shunt branch connected to the corresponding bus.

2.2.3.2 The measurement Jacobian matrix, H(x)

It can be seen in equation (2.3) that the Jacobian of the measurement function vector *h(x)* should be calculated. Depending on which measurements are used in the Jacobian matrix, it consists of the derivative of the functions relating those measurements to state variables with respect to state variables, i.e. voltage and angle here.
This can be done easily by taking the derivatives of equations (2.6) to (2.8) with respect to voltage V, and angle Θ.

2.3. PMU-Only State Estimation

As mentioned in Chapter 1, the introduction of PMUs has promoted many studies toward its application in power systems. Its ability to provide synchronous phasor measurements of voltage and current has made it possible to propose many power system applications faster, and with higher accuracy in real-time. This will lead to a substantial increase in the situational awareness in power networks.

Actually it was after the 2003 US-Canada blackout that the importance of PMUs to provide synchronized system data was realized. That one is among the largest blackouts during the history which affected more than 50 million people, and cost more than $4 billion for the American economy. Initially started by the disconnection of a line in Ohio, the postmortem report which was provided by Federal Energy Regulatory Commission to the US Congress in 2006, indicated that it was "caused in part by deficiencies by the system that monitors the electric grid and a lack of awareness of deteriorating conditions by the operators who monitor the system". The U.S.-Canada power system outage task force recommended the usage of time synchronized data recorders to all utilities in order to prevent similar events in the future [15]. In fact, if it was not due to insufficient situational awareness, the 2003 blackout would have been prevented in its initial stage of occurrence. This was the inception of moving toward the integration of wide-area monitoring into power systems. Wide-area monitoring systems gather data synchronously from PMUs using a common timing signal, usually from a GPS.

As described in Chapter 1 that there are proposed methods to use PMU measurements in state estimators. This section provides a brief introduction of PMUs, and then explains two of the most important works in the area of PMU-only state estimation.

2.3.1. Synchronized Phasor Measurements: An Introduction

It was in 1988 that Phasor Measurements Unit (PMU) was firstly introduced by Dr. A. Phadke and Dr. J. Thorp from Virginia Tech. It directly measures the electrical phasor, i.e. both its magnitude and phase angle. Fig. 2-2 is a good representation of a phasor for a sinusoidal wave.

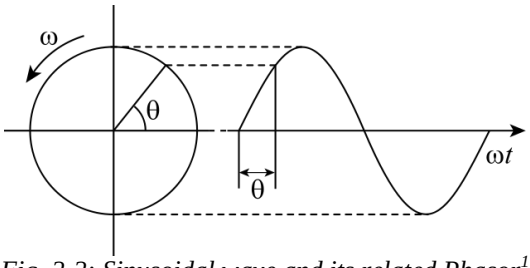


Fig. 2-2: Sinusoidal wave and its related Phasor[1]

[1] Image taken from Wikipedia: http://commons.wikimedia.org/wiki/File:Phasor.svg

In addition, synchronized phasors have the advantage of using the Global Positing System (GPS) to timestamp each measured snapshot of data. This is a huge advantage as previously network operators had to gather asynchronous data through ICCP protocol which would give them little information about the subsequent events and their timing. But GPS has made it possible to gather data from different parts of a power network, and then sort them to find the snapshots that occurred at the same time, called as "synchrophasors". The introduction of synchronphasors made the comparison of different measurements which happened at the same time possible, which is critical to find out the system condition. This will finally increase the situation awareness of the power network significantly. It is also very important that PMUs can telemeter data with speed as fast as 30-60 of samples per second, while previously the fastest rate of measurement telemetering was once a second.

The above mentioned characteristics of phasor measurement units make them extremely likely to be used in power system applications, and since their introduction many methods have been proposed how to exploit the advantages of PMU in power networks [3]. This is also true for state estimation and novel algorithms have been proposed on how to apply synchrophasors in state estimation. In coming sections, two of the most important works in this area are explained.

2.3.2. Linear State Estimation

The linear state estimation approach was proposed by A.G. Phadke, J.S. Thorp and K.J. Karimi in 1986 [4]. The idea behind the linear state estimation method is to measure the bus voltages and line currents phasors and create the estimation problem based on linear relation between complex voltages and currents. A very simple network is show in Fig. 2-3:

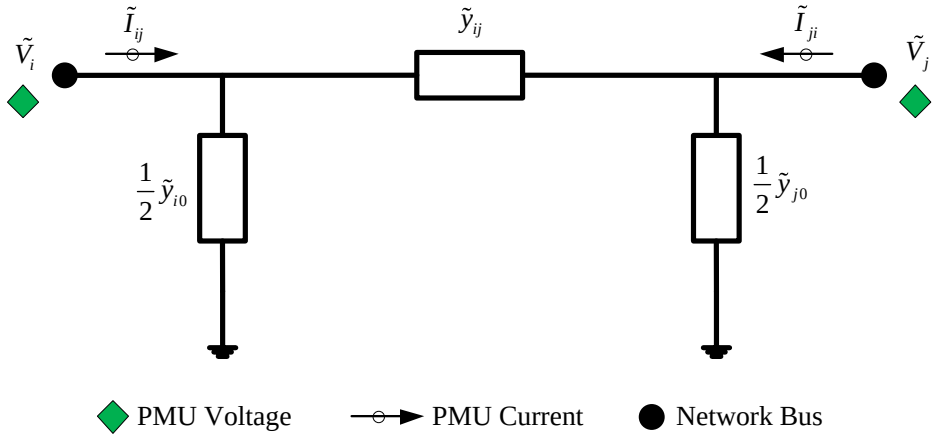


Fig. 2-3: Simple Network Model

The following equations can be written based on the network shown in Fig. 2-3:

$$\begin{aligned}\tilde{I}_{ij} &= \left(\tilde{y}_{ij} + \tilde{y}_{i0}\right)\tilde{V}_i - \tilde{y}_{ij}\tilde{V}_j \\ \tilde{I}_{ji} &= \left(\tilde{y}_{ij} + \tilde{y}_{j0}\right)\tilde{V}_j - \tilde{y}_{ij}\tilde{V}_i\end{aligned} \tag{2.9}$$

A bus admittance matrix *Y* is defined which consists of only the admittances of the lines with PMU measurements:

$$\tilde{Y} = \begin{bmatrix} \tilde{y}_{ij} + \tilde{y}_{i0} & -\tilde{y}_{ij} \\ -\tilde{y}_{ij} & \tilde{y}_{ij} + \tilde{y}_{j0} \end{bmatrix} = \tilde{y}A^T + \tilde{y}s \tag{2.10}$$

Here *A* is a current measurement-bus incidence matrix which is composed of *m* rows and *b* columns; *m* is equal to the number of available current phasor measurements and *b* is equal to the number of available voltage phasor measurements. In addition, *y* is a *(m×m)* diagonal matrix of series admittances, while *ys* is a *(m×b)* primitive matrix of shunt admittances. The complex currents can be written as:

$$\tilde{I} = \left(\tilde{y}A^T + \tilde{y}s\right)\tilde{V} \tag{2.11}$$

The measurement vector *z* consists of measured voltage and current phasors:

$$z = \begin{bmatrix} \tilde{V}_m \\ \tilde{I}_m \end{bmatrix} \tag{2.12}$$

Now, a linear matrix of measurement functions *h(x)* can be created relating the states, i.e. the voltage phasors to the complex measurements:

$$h(x) = \begin{bmatrix} U \\ \tilde{y}A^T + \tilde{y}s \end{bmatrix}\tilde{V} = \tilde{B}\tilde{V} \tag{2.13}$$

Here, *U* is the identity mtrix. Now the estimation problem can be formulated in the same way as (2.1):

$$z = h(x) + e \tag{2.1}$$

Here, the solution to (2.1) is given by:

$$\tilde{G}x = \tilde{B}^T Rz \tag{2.14}$$

Where *G* is the complex gain matrix equal to:

$$\tilde{G} = \tilde{B}^T R\tilde{B} \tag{2.15}$$

By computing the gain matrix, $\tilde{G}x = \tilde{B}^T Rz$ can be solved by performing triangularization and back-substitution. The upper triangular matrix should be stored.

Notice:

The state estimation solution by this method is obtained in rectangular format and should be mapped into polar coordination to be meaningful.

2.3.3. Phasor State Estimation

Phasor State Estimation is a method propsed L. Vanfretti and J. Chow as a PMU-Only based State Estimator in 2009 [5]. It requires a modest number of PMUs installed in the HV backbone of the power system. It has several advantages such as it can support a conventional state estimator so to provide additional reliability whenever SE is in function, and makes the HV network observable whenever the SE is not available. The aim is not to replace conventional state estimators, but to build a parallel and redundant state estimator that provides a wide-area visibility of large-scale networks by focusing on the HV backbone of the transmission system.

Additionally, the idea of exploiting redundancy to extrapolate information to buses which are not equipped with PMUs enables the method to perform angle bias correction. By angle biases we refer to a random shift in angles which occurs sporadically in a PMU. The source of such a kind of error is not exactly know; however, it can be due to some particular error in signal processing algorithm within the PMU itself, or error with time synchronization and internal clock synchronization with GPS. The Phasor State Estimation consists of many sections which each one is briefly discussed in the following sections.

2.3.3.1 Angle Error Modeling in Polar Coordinates

The angle error in a PMU measurement is shown in Fig. 2-4 [5]:

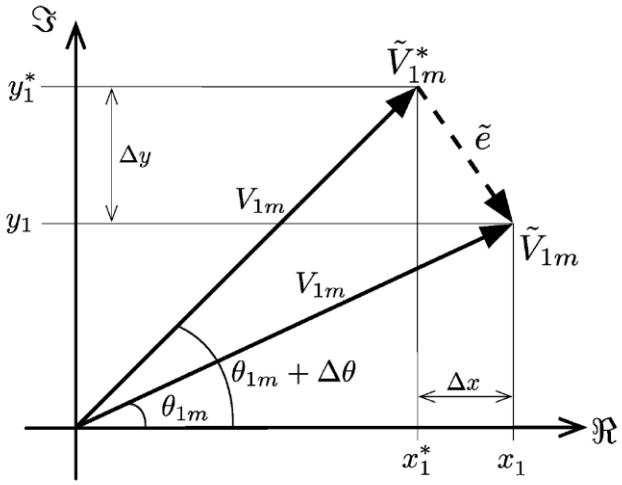


Fig. 2-4: Polar and Rectangular coordinate representation of an angular error in a PMU. Image taken from [5].

Although the rectangular coordinates model allows for a non-iterative solution, polar coordinates are more appropriate here as voltage magnitude and angle are mostly uncorrelated variables. In fact, rectangular coordinates don't allow modeling those variables independently. In Fig. 2-4:

- $\tilde{V}_{1m} = V_{1m}\varepsilon^{j\theta_{1m}}$ is the actual measured voltage phasor
- $\tilde{V}^*_{1m} = V_{1m}\varepsilon^{j(\theta_{1m}+\theta_e)}$ is the phasor measurement V_1 with an angle error of Θ_e

- $\tilde{e} = \tilde{V}_1 - \tilde{V}_1^*$ is the error phasor.

In polar coordinates, an angle error can be modeled with a linear angular unknown $\Phi = \Theta_e$ which is obviously not dependent on the phasor magnitude V_{1m}.

2.3.3.2 Measurement Model in Polar Coordinates

Fig. 2-5, shows a simple sample of an observable island in a power network [5]:

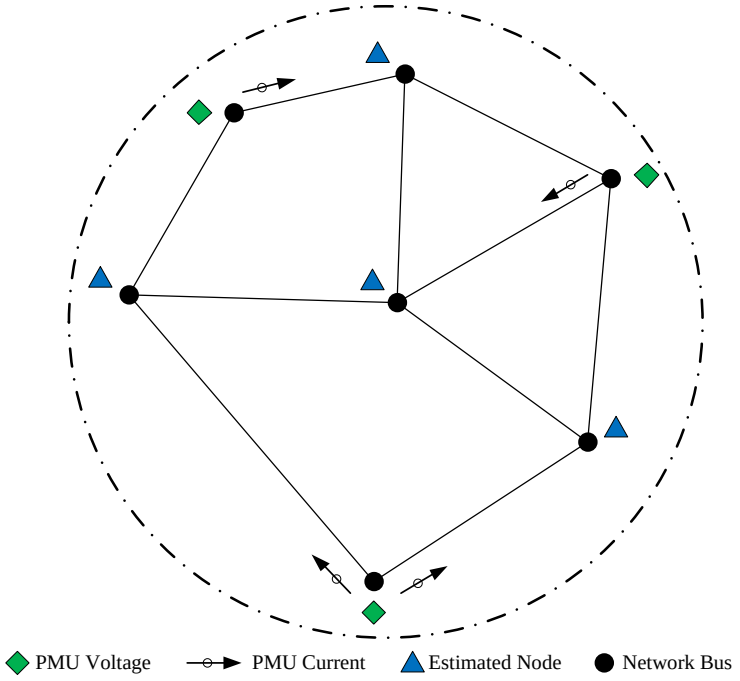


Fig. 2-5: PSE observable island

Having determined the observable island, the state vector is formed by gathering the magnitude and angle of the all voltages and currents, either those which are directly measured or those that are going to be estimated. Assume that in a system with N buses, N_1 buses have measured voltage phasors and N_2 buses are non-PMU ones. In addition, there are L lines which L_1 lines have measured current phasors and L_2 lines are unmonitored. The following matrices can be formed:

$$
\begin{aligned}
x &= \begin{bmatrix} V & I & \theta & \delta \end{bmatrix}^T \\
V &= \begin{bmatrix} V_1 & \cdots & V_{N_1} & V_{N_1+1} & \cdots & V_N \end{bmatrix}^T \\
\theta &= \begin{bmatrix} \theta_1 & \cdots & \theta_{N_1} & \theta_{N_1+1} & \cdots & \theta_N \end{bmatrix}^T \\
I &= \begin{bmatrix} \cdots & I_{ik} & \cdots \end{bmatrix}^T \\
\delta &= \begin{bmatrix} \cdots & \delta_{ik} & \cdots \end{bmatrix}^T
\end{aligned}
\qquad (2.16)
$$

At each PMU bus, the available measurements are voltage phasor and current. We pair each voltage and current measurements to its respective state:

$$V_i = V_{im} + e_{V_i}, \qquad \theta_i = \theta_{im} + e_{\theta_i} \tag{2.17}$$

$$I_{ik} = I_{ikm} + e_{Ik}, \qquad \delta_{ik} = \delta_{ikm} + e_{\delta_{ik}} \tag{2.18}$$

2.3.3.3 Network Model

Fig. 2-6, represents a very simple sample of network model [5]:

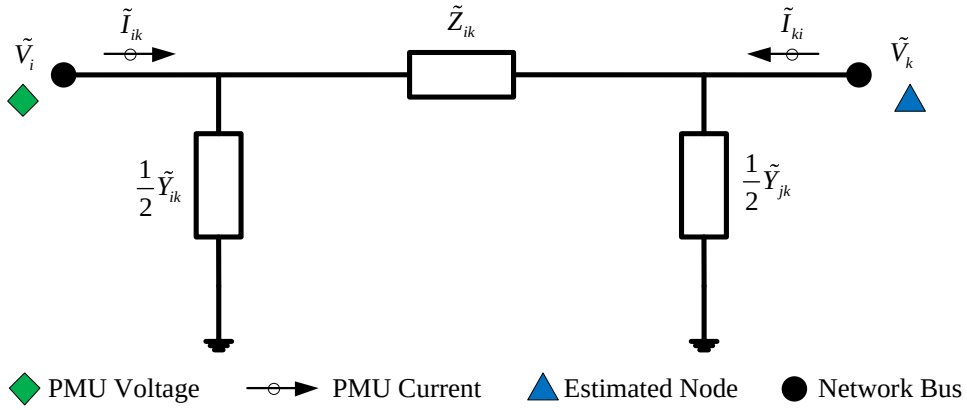


Fig. 2-6: PMU Bus-I circuit for the PSE network model

Using Fig. 2-6, the following equation can be derived:

$$\tilde{V}_k = \tilde{V}_i - \tilde{Z}_{ik}\left(\tilde{I}_{ik} - \frac{1}{2}\tilde{Y}_{ik}\tilde{V}_i\right) \tag{2.19}$$

Equation (2.19) is in complex form, so it can be divided in real and imaginary parts:

$$f_{ik} = \left(1 + \frac{1}{2}\tilde{Y}_{ik}\tilde{Z}_{ik}\right)\tilde{V}_i - \tilde{Z}_{ik}\tilde{I}_{ik} - \tilde{V}_k \tag{2.20}$$

$$f_{ikre} = \mathrm{Re}(f_{ik}) = 0, \qquad f_{ikim} = \mathrm{Im}(f_{ik}) = 0 \tag{2.21}$$

Equations (2.19) can be assembled into a 2L dimensional nonlinear vector f:

$$f = \begin{bmatrix} \cdots & f_{ikre} & f_{ikim} & \cdots \end{bmatrix}^T \tag{2.22}$$

2.3.3.4 PSE Solution-WLS Formulation

The objective of PSE solution is to satisfy the network equations and at the same time minimizing the errors in the measurement equation [5]:

$$\min_{x} \; q(x), \quad \text{subject to: } f = 0 \tag{2.23}$$

$$q(x) = \frac{1}{2}\left(\left\| W_V e_V \right\|^2 + \left\| W_I e_I \right\|^2 + \left\| W_\theta e_\theta \right\|^2 + \left\| W_\delta e_\delta \right\|^2 \right) \tag{2.24}$$

As said in previous section, *f* is a nonlinear function of voltage and current magnitudes and angles; considering the equality constraint *f = 0* to the objective function *q(x)* is given by:

$$q'(x) = q(x) + \frac{1}{2}\left\| W_f \, f \right\|^2 = \frac{1}{2}\left\| Wh(x) \right\|^2 \tag{2.25}$$

Here, W_f is a diagonal matrix with unity weights:

$$h = \begin{bmatrix} f^T & e^T \end{bmatrix}^T, \quad e = \begin{bmatrix} e_V^T & e_I^T & e_\theta^T & e_\delta^T \end{bmatrix}^T \tag{2.26}$$

$$W = block - diag\left(W_f \quad W_V \quad W_I \quad W_\theta \quad W_\delta \right) \tag{2.27}$$

W matrices are called weighting matrices, and calculated depending on type of the measurement equations. Except for some current magnitudes, most of the other weights are set to unity:

$$W_I = diag\left(\cdots, \min\left(1, \frac{1}{I_{ikm}} \right), \cdots \right) \tag{2.28}$$

Considering equations (2.25) to (2.27), it can be seen that the constrained WLS problem of (2.23) can be transformed to the unconstrained problem of:

$$\min_{x} \; q'(x) \tag{2.29}$$

Notice:

When the number of constrained equations are equal to number of unknowns, i.e. $L_1 = N_2$, there is a unique solution and the measurements error are taken to be zero. In the case that $L_1 > N_2$, the WLS solution gives the best fit.

As mentioned in 2.2, the WLS is an iterative solution that uses the Gauss-Newton algorithm and the objective is to find Δx in each step until a convergence constraint is satisfied. Here, Δx is calculated using the following equation:

$$\Delta x = -\left(H\left(x_c \right) \right)^{-1} \left(WJ\left(x_c \right) \right)^T h\left(x_c \right) \tag{2.30}$$

Where *J* is the Jacobian of the following form:

$$J = \left[\begin{array}{c|c|c|c} & \multicolumn{3}{c}{\frac{\partial f}{\partial x}} \\ \hline U_V & 0 & 0 & 0 \\ \hline 0 & 0 & U_I & 0 \\ \hline 0 & U_\theta & 0 & 0 \\ \hline 0 & 0 & 0 & U_\delta \end{array}\right] \tag{2.31}$$

In (2.31) U is an identity matrix.

2.3.3.5 Extension for Angle Correction

With some slight modification, the PSE algorithm can be able to detect and fix PMU angle biases. First the measurement model should be updated as follow:

$$\begin{aligned} \theta_i &= \theta_{im} - \phi_i + e_{\theta_i} \\ \delta_{ik} &= \delta_{ikm} - \phi_i + e_{\delta_{ik}} \end{aligned} \tag{2.32}$$

The angles bias terms form a vector as following:

$$\phi = \begin{bmatrix} \phi_2 & \phi_3 & \cdots & \phi_{N_1} \end{bmatrix}^T \tag{2.33}$$

Notice:

Φ_1 is not included in the angle bias vector as bus 1 is the reference bus.

The WLS problem of (2.29) can now be transformed to the new form of:

$$\min_{x_\phi} \; q'(x_\phi) \tag{2.34}$$

Here x_Φ is the modified state vector:

$$x_\phi = \begin{bmatrix} x^T & \phi^T \end{bmatrix}^T \tag{2.35}$$

The Jacobian matrix is also modified:

$$J_\phi = \left[\begin{array}{c|c} J & \frac{0}{\partial e(\phi)/\partial \phi} \end{array}\right] \tag{2.36}$$

Notice:

$\partial e(\phi)/\partial \phi$ is sparse and consisted of ones and zeros.

2.4. Chapter Summary

This chapter has given an introduction to State Estimation in power systems. Firstly, conventional state estimators were fully explained, and the problem formulation and measurement equations were developed. In addition, the concept of PMU and PMU-only based state estimation was explained. Two common algorithms in PMU-only state estimation were also explained; sufficient details were provided about these two algorithms, and advantages of one over the other have been briefly discussed.

Chapter 3: Network Topology Processor

3.1. Introduction

Network topology processing is a crucial step in any state estimator. Using breakers status and network connectivity data, it determines the network topology. This determination is done by transforming the bus-section/switching-device model of a power network to a bus/branch model. An example of such a transformation is provided in Fig. 3-1 and Fig. 3-2 [6]:

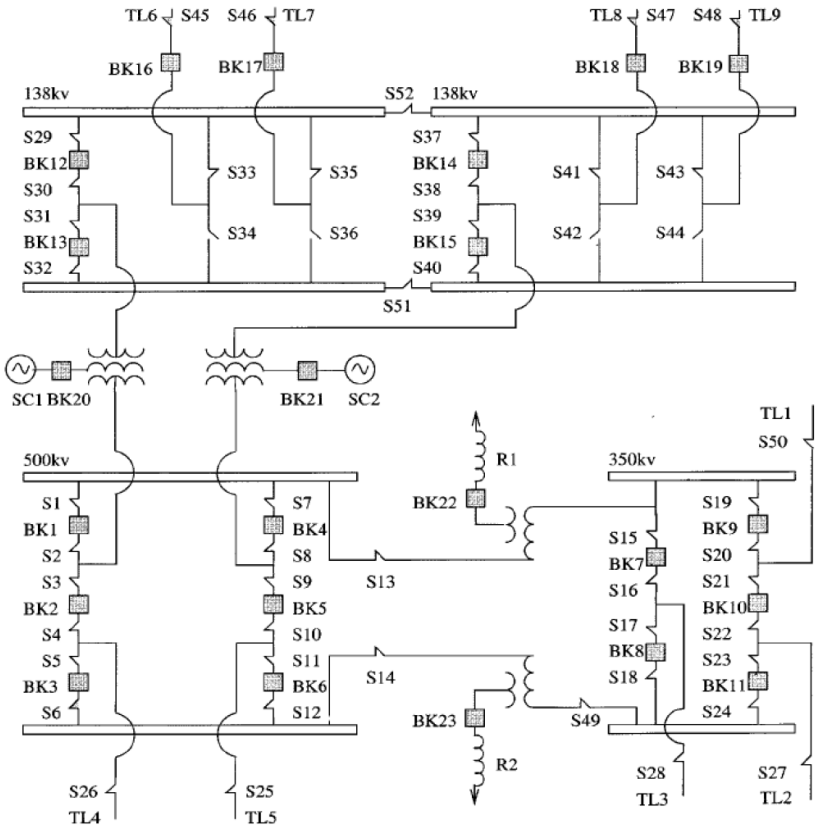


Fig. 3-1: Bus-section/Switching-device model taken from [6]

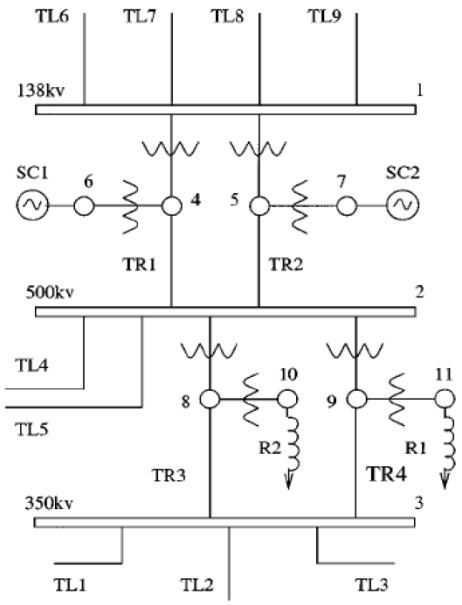


Fig. 3-2: Bus-section/Switching-device model

To perform this kind of transformation, many steps need to be defined in a topology processor. This section fully explains the steps that should be executed inside a topology processor.

3.2. Processing of Input Data

The input data to any topology processor consists of pre-defined constant connectivity parameters and breakers status. By connectivity parameters we refer to how the network components are connected to each other if all of the breakers in the system are closed. The breakers statuses are in binary form (either 1 or 0) representing closed and open positions respectively.

Traditionally, data is telemetered through analog transmitters. However, the arrival of PMUs made it possible to send the breakers status digitally with a much higher rate compared to telemetry solutions. In addition, the measurement of line currents by PMUs has led to proposal of new algorithms to perform the topology processing. In the case that the breaker statuses are going to be used as an input to topology processor, the algorithm does not change either using PMU or traditional transmitters; however, it is possible to update the topology with a higher rate using the PMU data as an input.

3.3. Substation Analysis

At the first glance, topology processing may seem to be an easy task; an open breaker represents an open line. However, this is far from reality. In a real power system network, there are lots of lines and substation which are connected to each other in an

extremely complex way. In fact, it is not like that whenever a breaker is opened a line is disconnected. This is due to the existence of different substation configurations. Generally, there are 6 main substation configurations; other configurations are a modification to these 6 main ones. Fig. 3-3 to Fig. 3-8 show these 6 main configurations [24]:

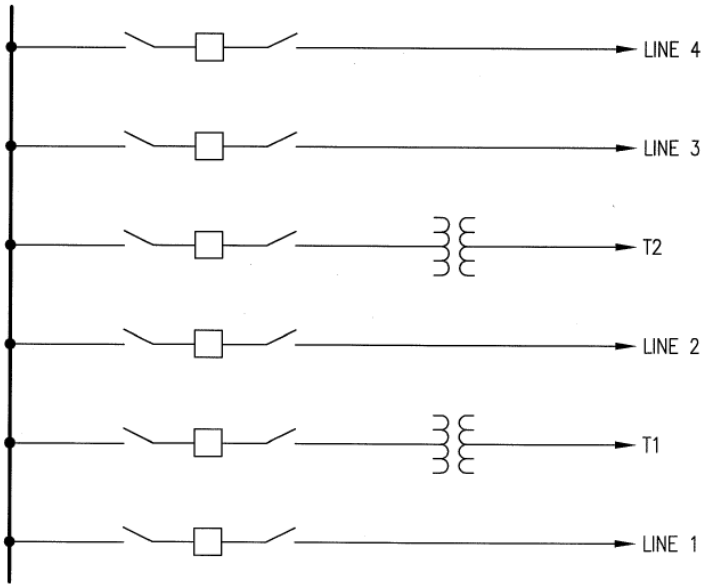


Fig. 3-3: Single Bus Configuration taken from[24]

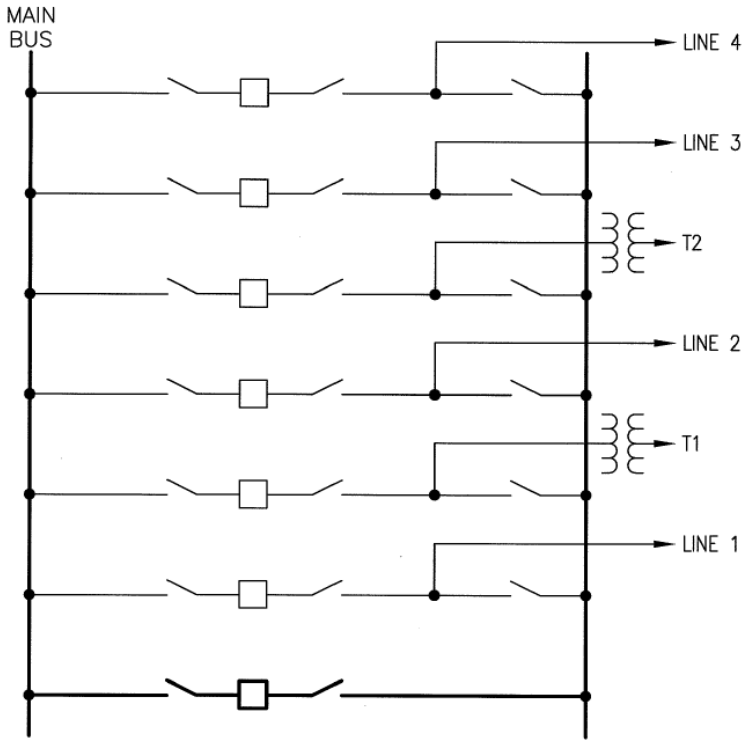


Fig. 3-4: Main & Transfer Bus Configuration taken from [24]

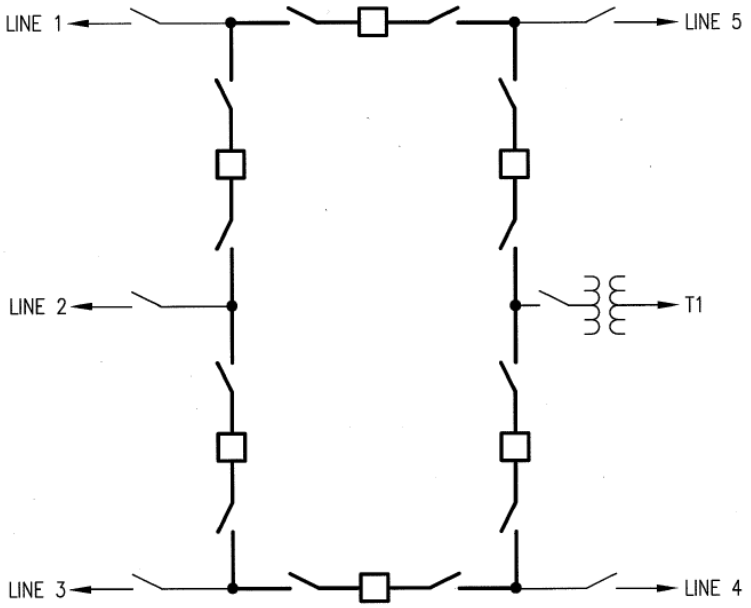


Fig. 3-5: Ring Configuration taken from [24]

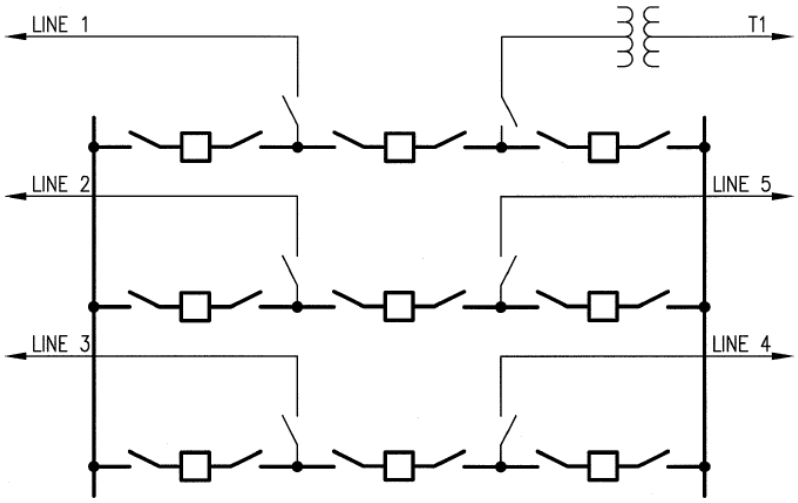


Fig. 3-6: Breaker and a Half Configuration taken from [24]

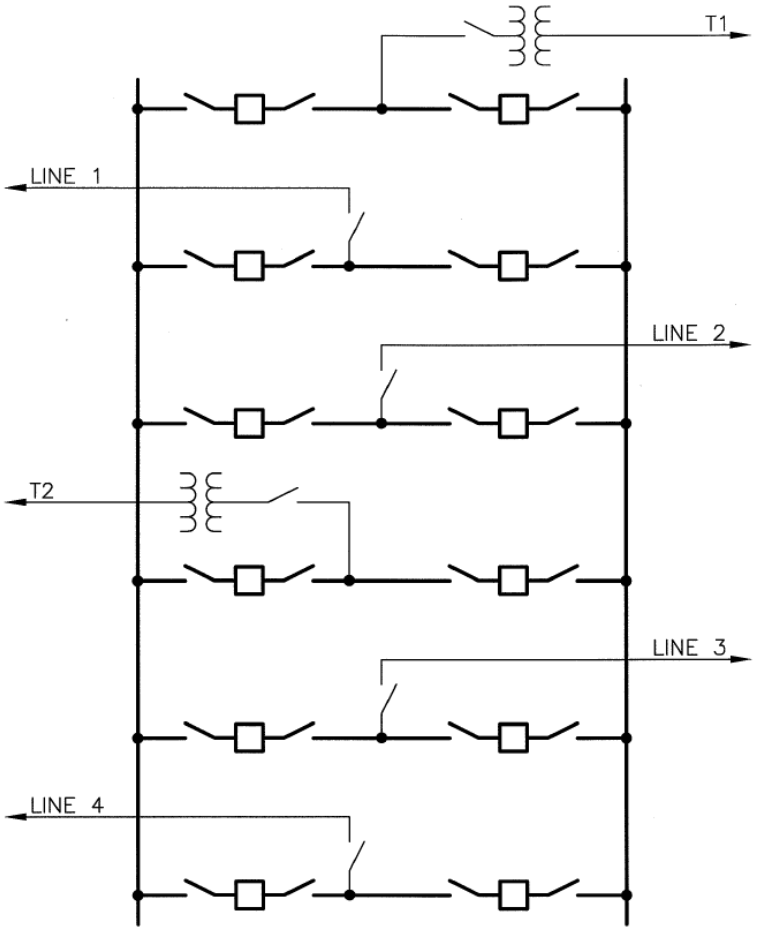


Fig. 3-7: Double Bus-Double Breaker Configuration taken from [24]

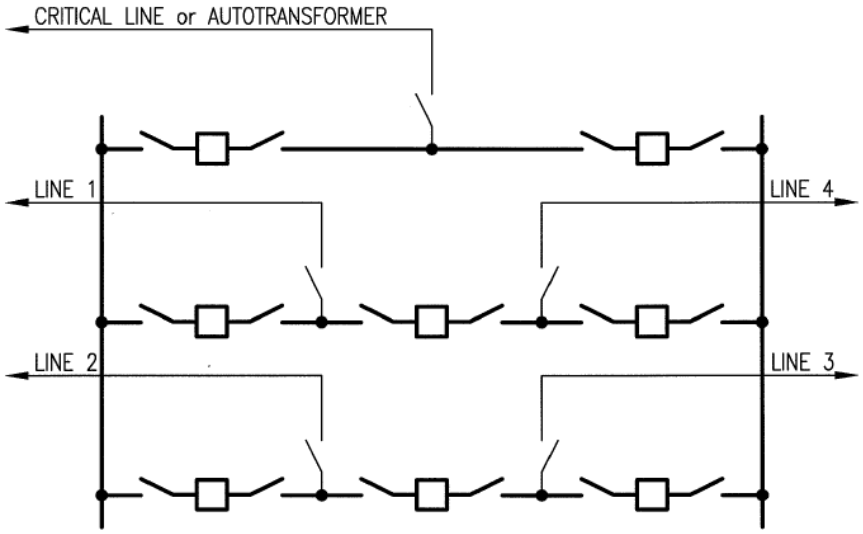


Fig. 3-8: Combination of Breaker and a Half and Double Bus-Double Breaker Configuration taken from [24]

Considering Figs. 3-3 to 3-8, it can be seen that in most of the configurations an open breaker does not necessarily mean an open line or circuit. In addition, there is a possibility that a one single substation with a specific configuration is split to two or more substations with other configurations. Similarly, there may be cases in which two different substation merge together and form a new one. The task of substation analysis is to completely consider the substation configurations in a line connectivity analysis, and to detect the possible splitting/merging of substations. This is sometimes very hard due to existing complex configurations. In addition, the lines connected and disconnected to a substation should be carefully scrutinized as an open breaker in a substation with a configuration other than single bus-breaker does not always mean that its corresponding line is open.

3.4. Islanding Analysis

There may be times in a power system network when an interconnected network is split into two or more separated areas. This may occur due to breaker switching that leads to an area having no connected line to the other part of the system. This event is usually called islanding, when a unified network is separated to two or more different islands. One of the main tasks of a topology processing engine is to detect if islanding has occurred in some part of the network. Reversely, it should be able to detect if some separated islands have merged together and formed a new island. In addition, automatic numbers should be assigned to the possible new splitting/merging islands.

3.5. Energization Analysis

This part is in charge of identifying and reporting energized, de-energized and grounded islands. Energized islands are those which have at least one operating generator. De-energized islands are those ones that have no working generator, but are not grounded; they are simply disconnected parts of the network. Grounded island is referred to an island which has no on-line generator, and at the same time is connected to the ground. This part is also of high importance as those parts of a network which are not energized would not participate in any network analysis. Therefore there is no need to formulate the equations relating to those parts that are either de-energized or grounded.

3.6. Chapter Summary

This chapter has provided a descriptive explanation of network topology processors. First the concept of topology processing is introduced and then basic steps involved in a topology engine are explained one by one. Different substation configurations are shown in this chapter as they play an important role in defining how efficient a topology processor is. This chapter contains no mathematical or specific algorithm about how to perform a topology processing, as this part is fully investigated and discussed in chapter 5 of this thesis.

Chapter 4: Test System Modeling in SimPowerSystems for Real-Time Simulation

4.1. Introduction

As said in Chapter 1, one of the tasks of this thesis is to build, implement, and preform real time simulation for the IEEE Reliability Test System 1996 [14]. This is done using the eMegaSim Opal-RT real time simulator [27] which is owned by SmarTS Lab [17], KTH Royal Institute of technology. As this device's simulation platform is the SimPowerSystems Toolbox [26], the model should be implemented in MATLAB Simulink, and then modified in order to be able to run in real-time using RT-LAB software [28] which is designed by Opal-RT to run models in a real-time target.

This chapter is dedicated to introducing the test system used, and a brief overview on its Simulink model. In addition, the designed test scenarios are introduced and finally the test results are demonstrated. Details of Simulink modeling and RT-LAB modification are not discussed as they are out of the scope of this thesis. For more information regarding SimPowerSystems modeling and RT-LAB modification, please refer to [25].

4.2. IEEE Reliability Test System 1996

The single line diagram of IEEE Reliability Test System which is modeled in SimPowerSystems is shown in Fig. 4-1

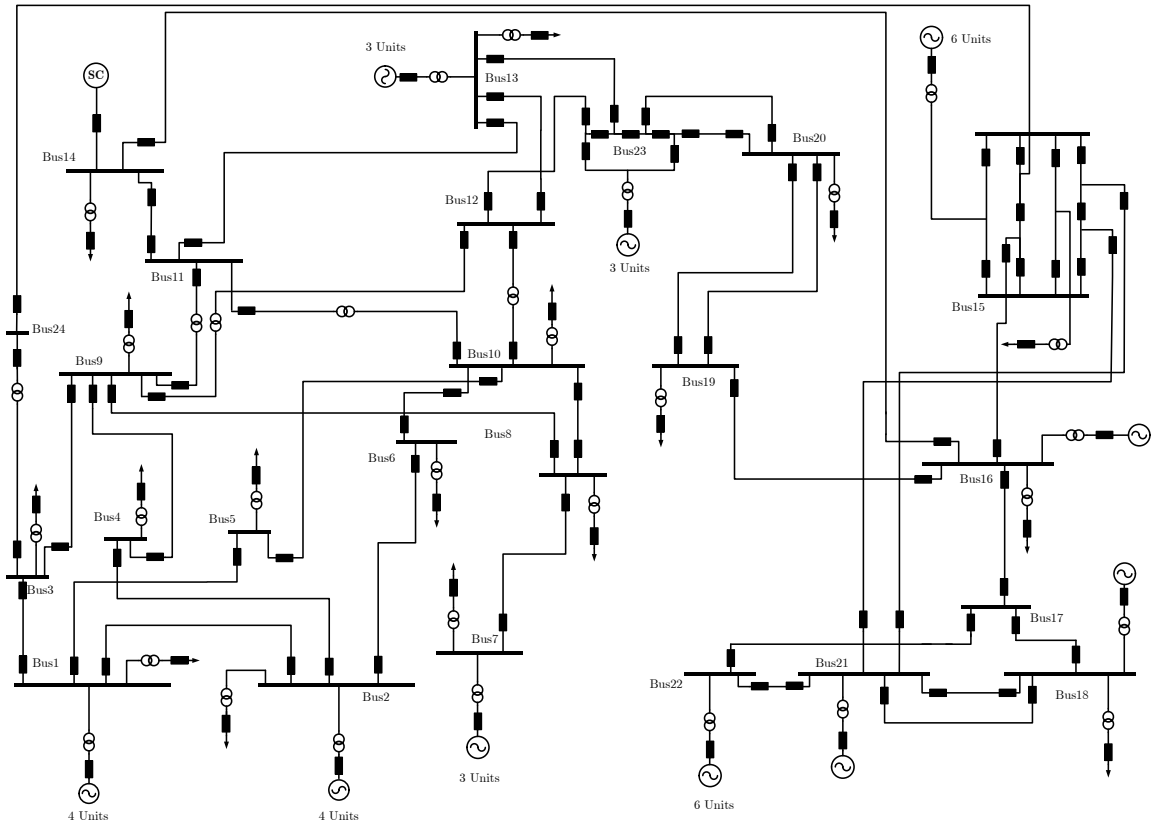


Fig. 4-1: IEEE Reliability Test System 1996

The figure shows various power system components connected together to form a power system model. Some details about the model used in this simulation are:

- It is comprised of 10 synchronous generators.
- It consists of 24 substations, with configurations such as Single Bus Single Breaker, Double Bus Double Breaker, Breaker and a Half, and Ring.
- There are 119 breakers in this system.
- There are 66 lines and 34 buses in this system.

In order to make the system compatible with the eMegaSim Opal-RT real time simulator, there is a need to split it into several subsystems. Actually, the Opal-RT real time simulator owned by SmarTS Lab, has two units, each one with 12 cores. These cores provide the possibility to run the model simulation in parallel, i.e. each subsystem is running on a different core. Therefore, the model can be executed in real-time as the

computation burden is divided between several cores. Another important modification is that there should be always a

subsystem containing all the scopes and controllers in the model. This subsystem is called “Console” and is the interface between the model running in the real-time targets and the user. All the monitoring devices as well as controlling ones should be located in the Console, and proper communication should be established between Console and other subsystems. In this thesis, the model which is shown in Fig. 4-1 is divided into 9 subsystems as below:

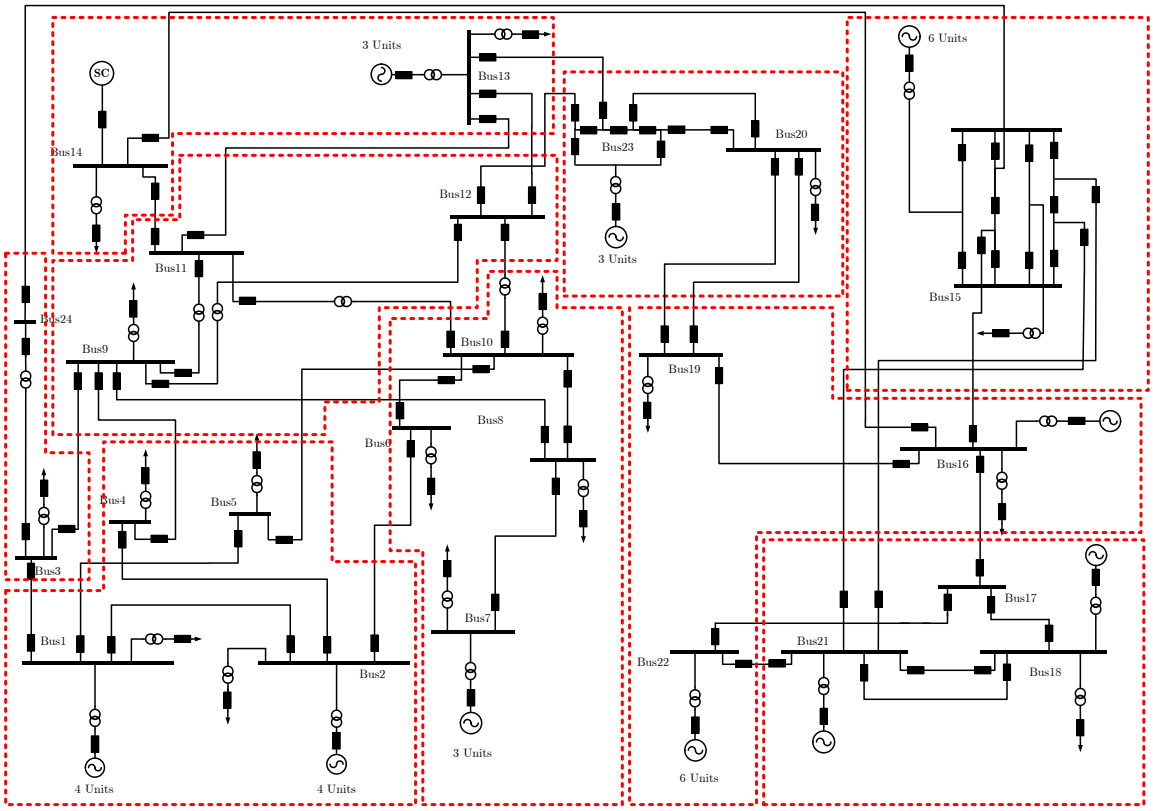


Fig. 4-2: Original Model divided into 9 subsystems

The connection between different subsystems is established using the existing transmission lines (PI line model in SimPowerSyetms). These 9 subsystems, along with the Console subsystems, form the final 10 subsystems of simulated model in Simulink. Fig. 4-3 to Fig. 4-6 demonstrate some examples of the implemented model.

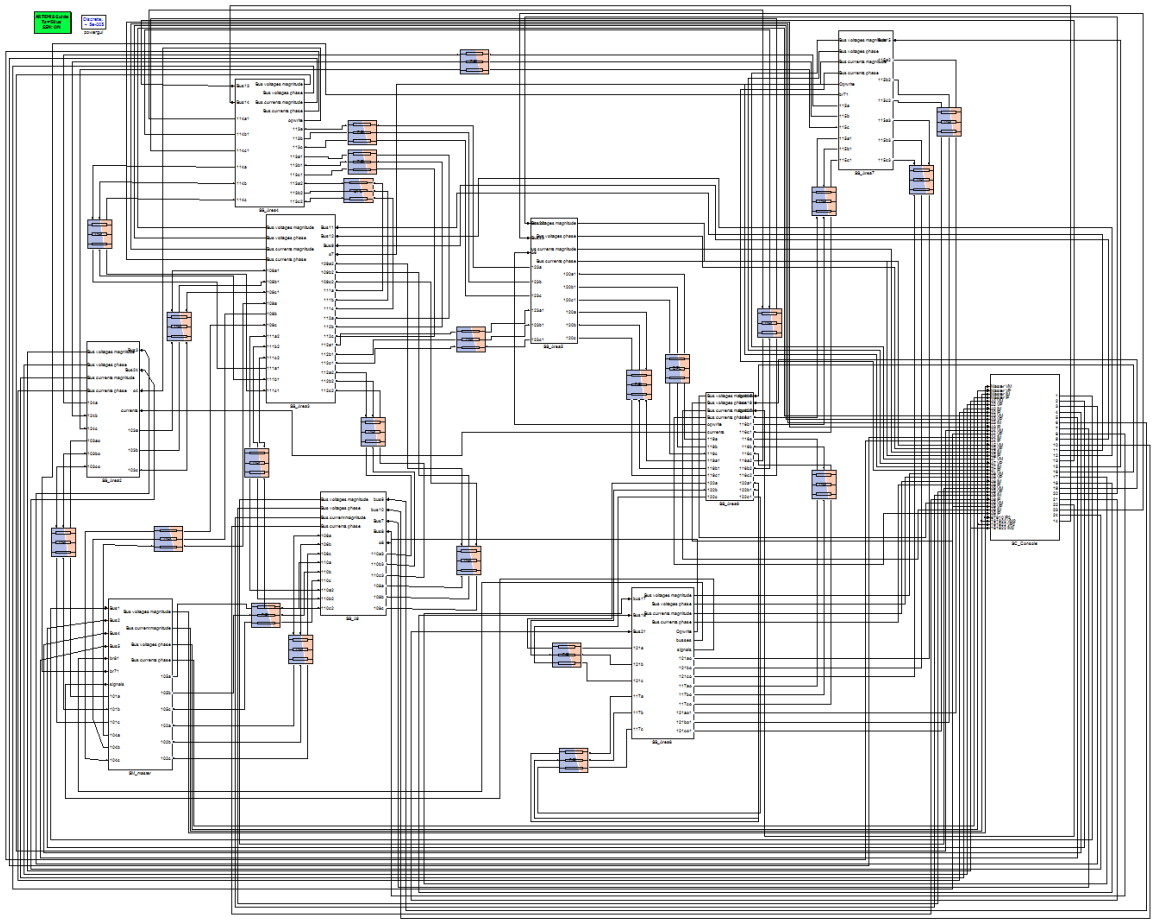


Fig. 4-3: Model Implementation in Simulink

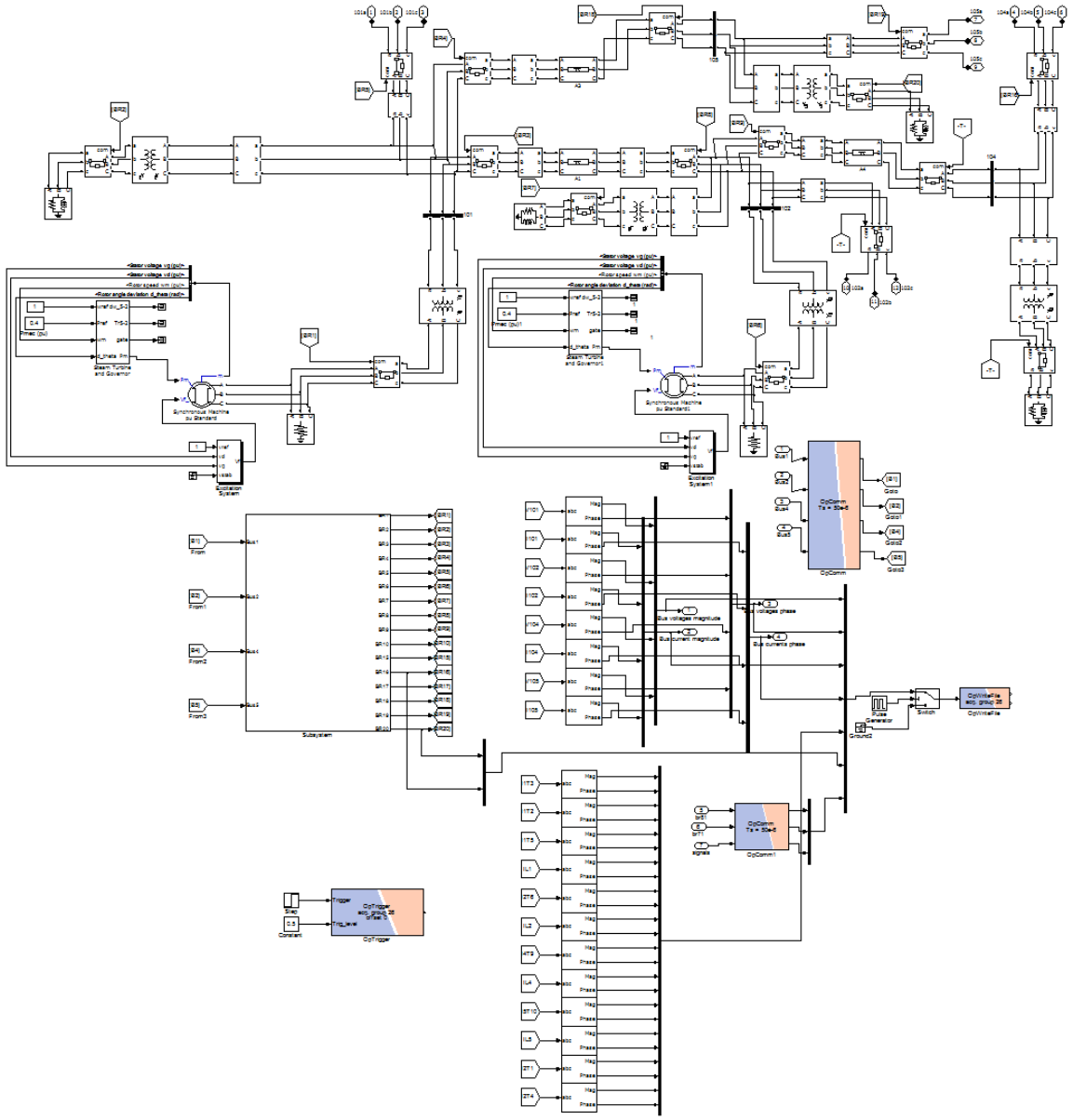


Fig. 4-4: Area 1

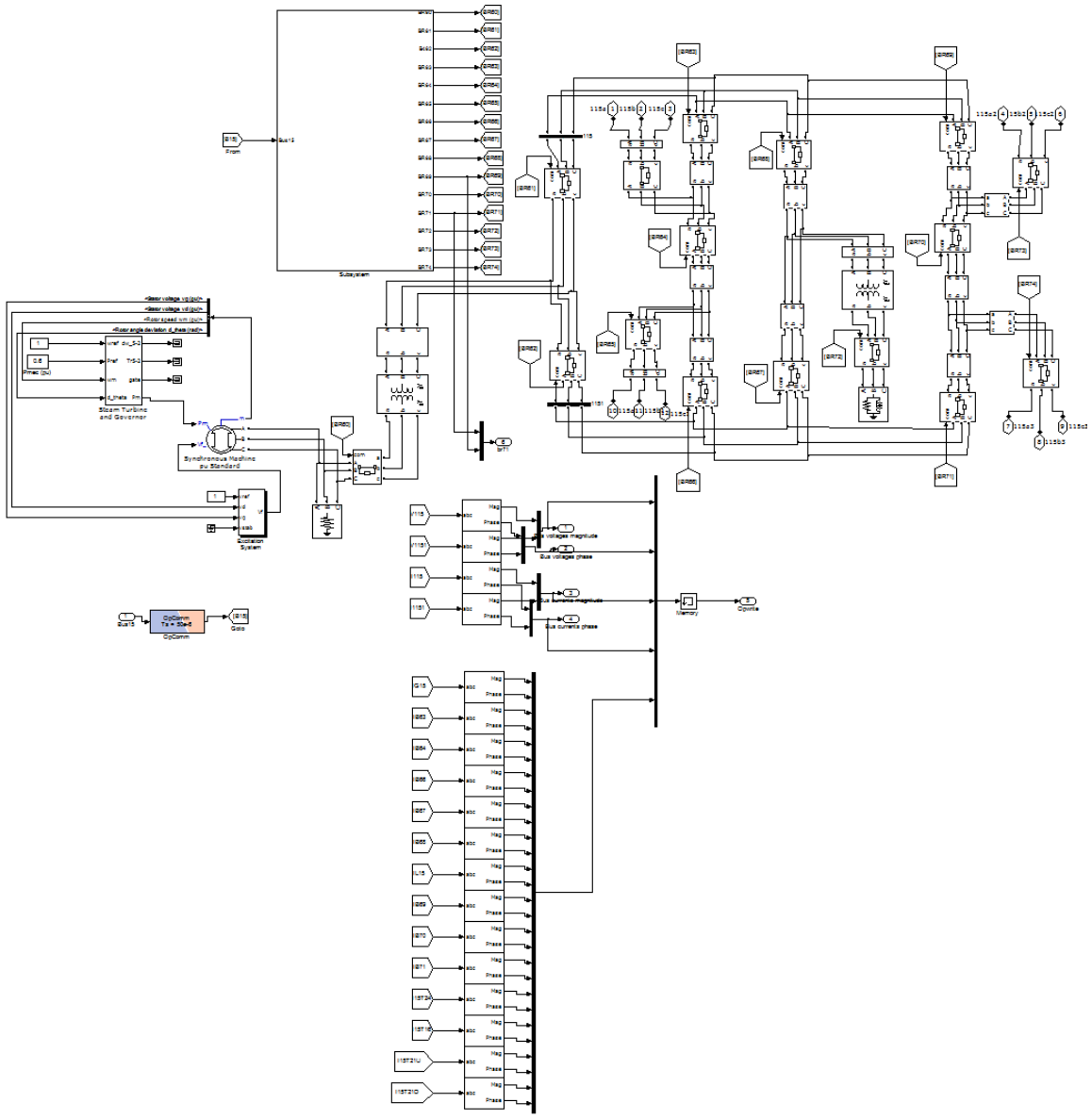


Fig. 4-5: Area 7

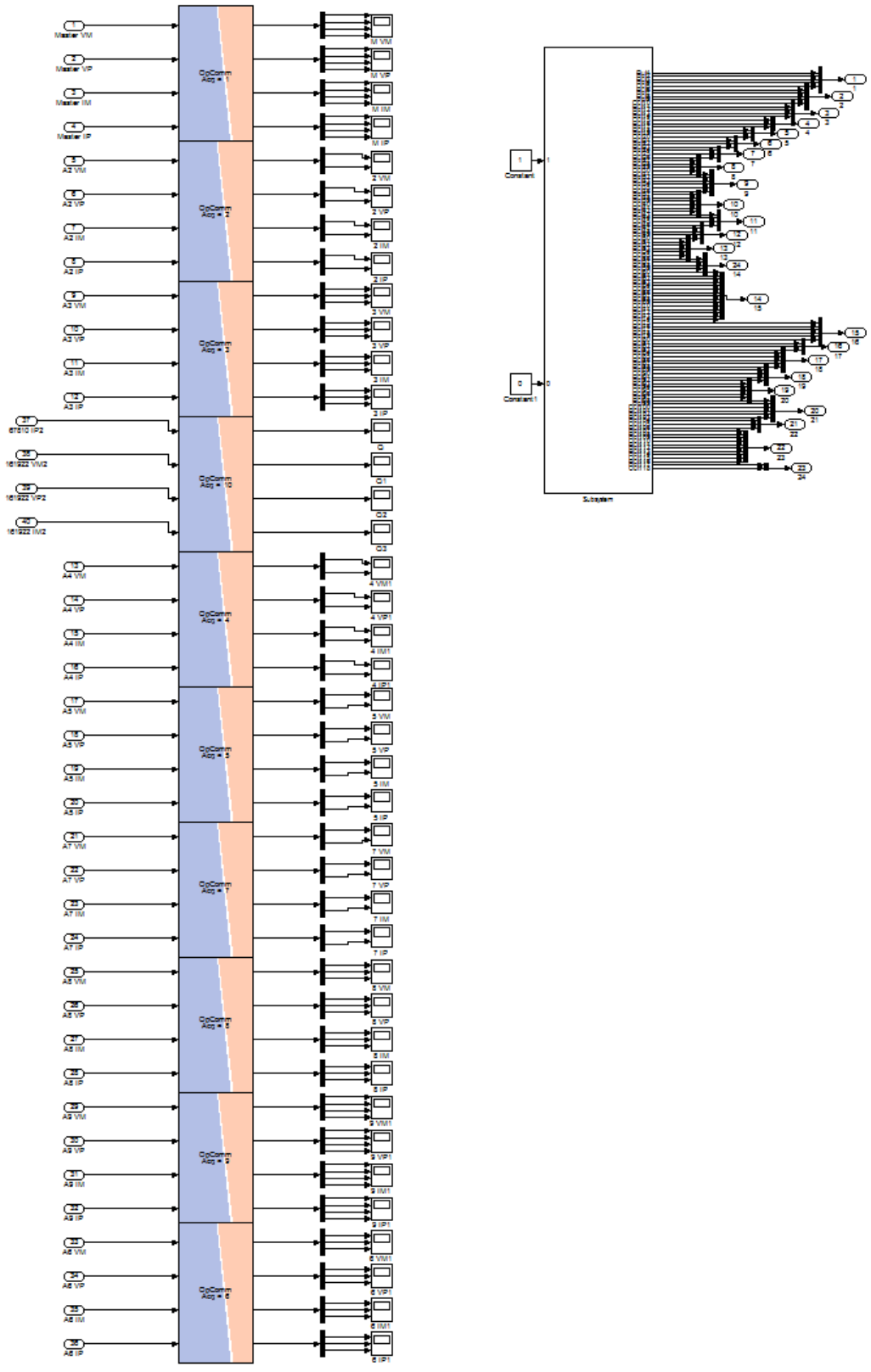


Fig. 4-6: Console

4.3. Data Logging

By Data logging, we refer to the action of saving measurements from a simulated model into m-Files for further plotting, analysis, or applications. Actually, the aim of simulating the model in real-time in this thesis is to obtain some measurements which are similar to measurements obtained from real PMU. So there is a need to read the voltage and current through the system and save them. After the RT-Lab software is installed, it will automatically add the RT-Lab library to Simulink. This Library consists of some blocks which are used for real-time simulator. One of those blocks is OpWriteFile which is specifically used for data logging:

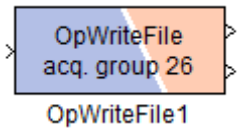
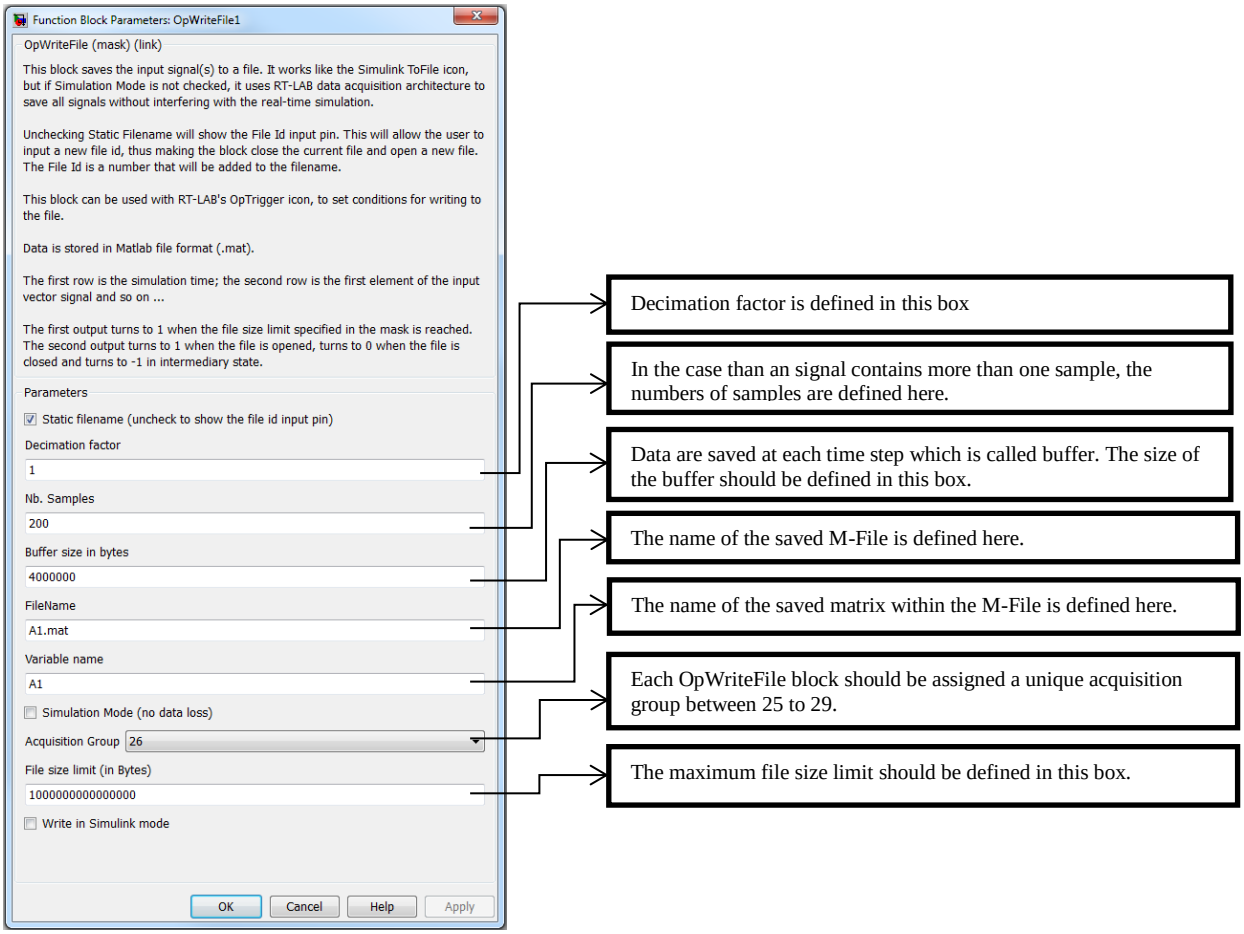


Fig. 4-7: OpWriteFile

The block reads the input signals and save them to an m-File with the same name as specified in the block parameters. The m-file is located in a folder corresponding to the same subsystem in which the block is located. The block reads the input signal for each time step and save the corresponding value in a matrix. The first row of this matrix saves the measured simulation time corresponding to the current time step.



Notice:

As it can be seen from the acquisition group box, the number of OpWriteFile blocks that can be used is limited to 5. As there is 9 subsystems in the simulated model, measurements should be sent from 4 subsystems to other 5 with OpWriteFile block.

The aim of data logging in this situation is to create some scenarios, save the voltage and current phasors and then used the saved data as it was measured data from PMUs. However, saving data is not as easy as it appears. The problem is that the number of measured values is high for an OpWriteFile block. Totally, there are 331 measured values; the best possible way to assign these measurements is to divide them almost equally between 5 different OpWriteFile blocks. Actually in the real model the least number of measurements assigned to an OpWriteFile block is 63, while the most is 77. This number of measurements is considered to be high for an OpWriteFile block to handle, and several problems will occur. First of all, the size of the file saving the data will be so high, e.g. after saving data for less than a minute the file would be in order of several Gigabytes. This will cause a file crash. Actually the file size limit that is in the block properties does not work properly in practice, and the file will crash if the size is greater than around 2 Gigabytes. In addition, even if the file does not crash, it can't simply save such a high amount of measurements as fast as ever time step, and it will skip to save data for other time steps randomly. This is also a major issue as there are 5 different OpWriteFiles in the simulation model, and each one shows its own random saving behavior which causes the saved measurements of the whole network not to be synchronized; this is important as we want the save data to be similar to real PMU data. In order to solve this drawback, a smart solution is applied as following:

- OpTrigger block: An typical OpTrigger block is shown in Fig. 4-8:

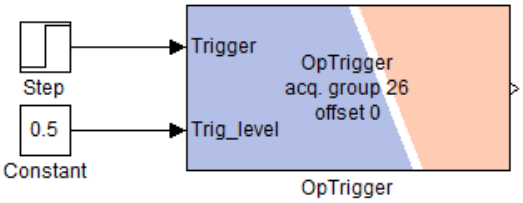


Fig. 4-8: OpTrigger

As it can be seen from the figure, this block has two inputs, Trigger and Trig_level. Instead of saving data from the beginning of the simulation, this block makes the user able to trigger data logging at a specific simulation time. The data logging will start whenever the input signal Trigger is greater than the Trig_level; therefore, it is possible to start saving data after the system has reached its steady state completely, and avoid saving unnecessary data of transients at the start of the simulation.

- Designing of a sampler: In addition to implementing an OpTrigger block, a data sampler is also designed and used in this simulation. Fig. 4-33 shows an overview of this data sampler.

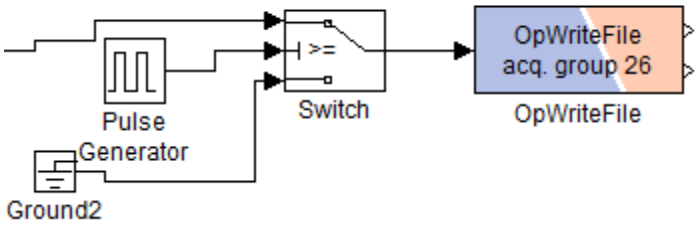


Fig. 4-9: Data Sampler

This simple data sampler consists of a switch, and a pulse generator. The switch has three different inputs: whenever the second input signal is greater than a predefined value, the switch passes the first input, otherwise it passes the third one. The strategy here is to design a scheme which passes data to the OpWriteFile block just 50 times per second (same as the rate of a typical PMU). This will result in a substantial decrease in the saved file size, and help to continue data logging for a larger simulation time. The pulse generator here produces 50 pulses per second, and the pulse width is 0.25% of the period which is equal to 50 micro second (same as the time step). The defined parameters of the Pulse Generator is shown in Fig. 4-10:

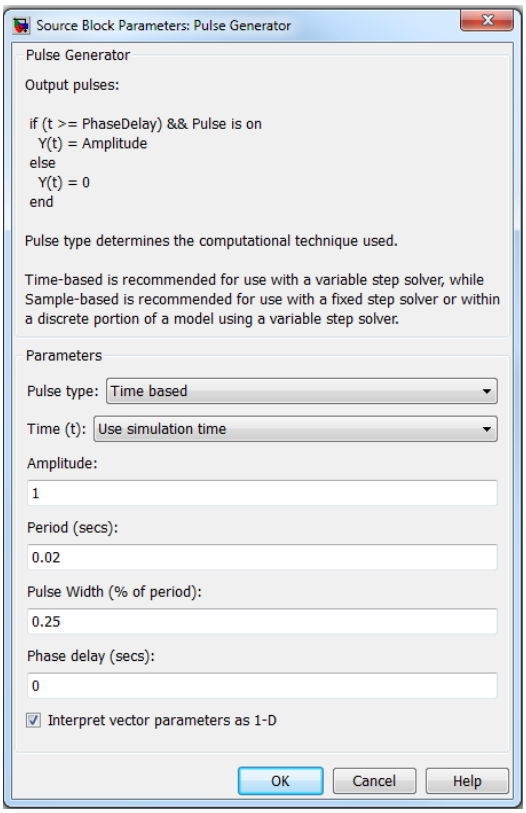


Fig. 4-10: Pulse Generator Parameters

After data logging is performed for a specific test scenario, there are 5 different saved files containing the measurements in the form an $n \times m$ matrix, which n is the number of saved snapshots, and m is the number of saved measurements. Note that the first row is always saved simulation time corresponding to the saved snapshot. To extract the saved data in this simulation, a simple MATLAB code is written. A part of this code is shown in Fig. 4-11:

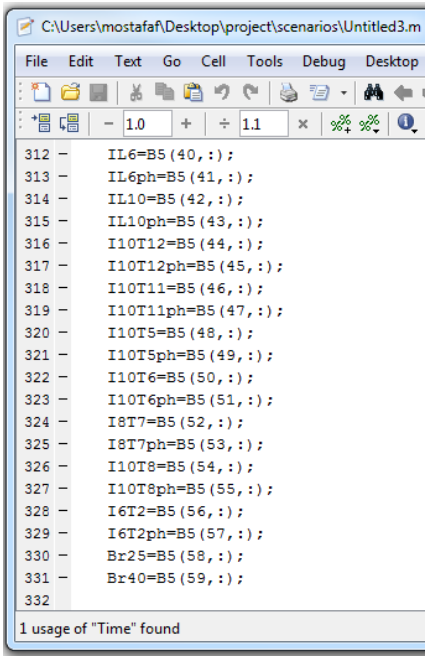


Fig. 4-11: MATLAB Code

To write this simple code, the order of saved measurements which were an input to their corresponding OpWriteFile block are investigated, and the rows of the saved matrix are extracted respectively with a name corresponding to the saved mesasurements. For example, if the 40th input measurement to an OpWriteFile block is the voltage of substation 6, this will be saved as the 41st row of the stored matrix (the first row is always the simulation time). Therefore, the 41st row of the saved matrix can be extracted and saved to a separate vector named as V6. The whole extracted rows can be saved then in a separate file corresponding to the performed test scenario.

4.4. Designed Test Scenarios

As it can be seen in Fig. 4-1, the test system is large and consists of 199 breakers, 24 substations with different configurations and 66 lines. This will provide the opportunity of performing many switching scenarios. Fig 4.12, show a larger scheme of the test system with the components being numbered:

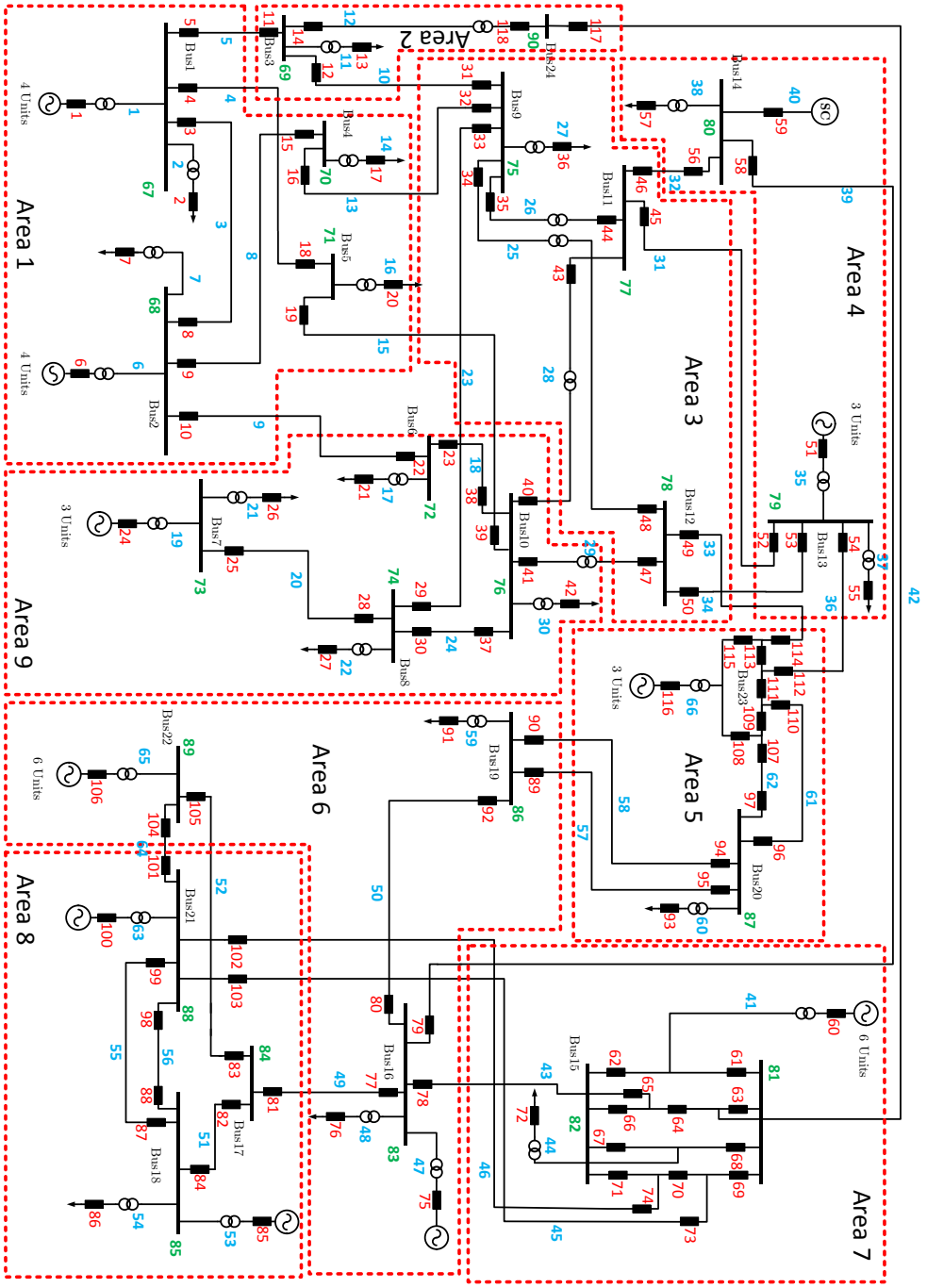


Fig. 4-12: Test System

Specifically, there are 10 designed test scenarios as follow:

1- Breaker 117 is opened.
2- Breaker 20 is opened.
3- Breaker 81 is opened and closed after 5 seconds.
4- Breaker 71 is opened.
5- Breaker 111 is opened and closed after 18 seconds.
6- Breaker 89 and 90 are opened at the same time.
7- Breaker 25 is opened and closed after 17 seconds.
8- Breaker 16 is opened and then breaker 84 is opened, and then closed at the same time.
9- Breaker 40 is opened and then breaker 43 is opened, then 40 is closed and then 43 is closed.
10- Breaker 109 is opened and then 111 is opened, and then 111 is closed and finally 109 is closed.

As said previously, there are 331 measured voltage and current phasors for each test scenarios, so obviously it is not possible to present all of the results of each test scenario. However, for each case one of the voltages profile is shown in the following figures, where the top plot corresponds to a voltage magnitude at a substation, and the second plot is the breaker status.

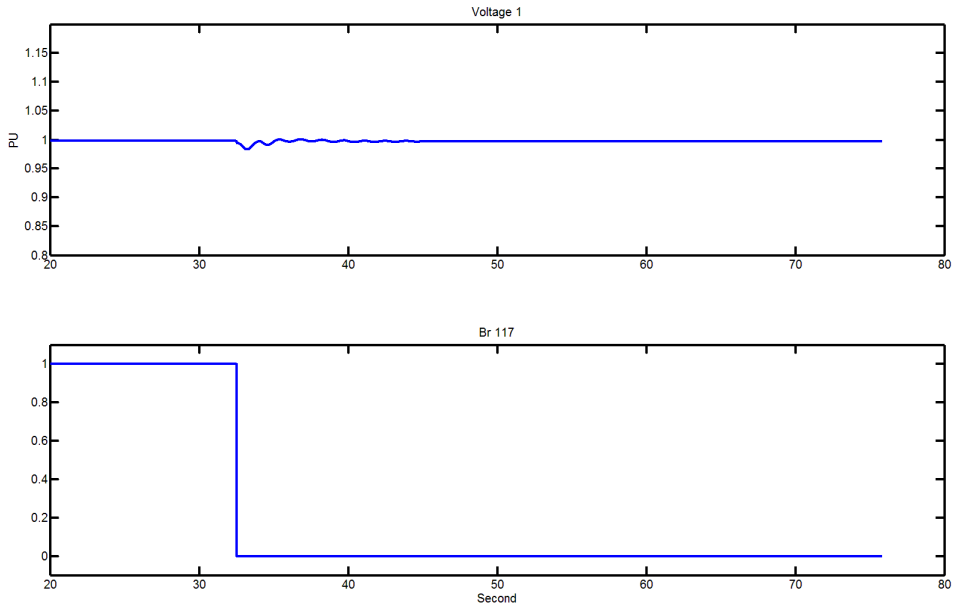


Fig. 4-13: Scenario 1

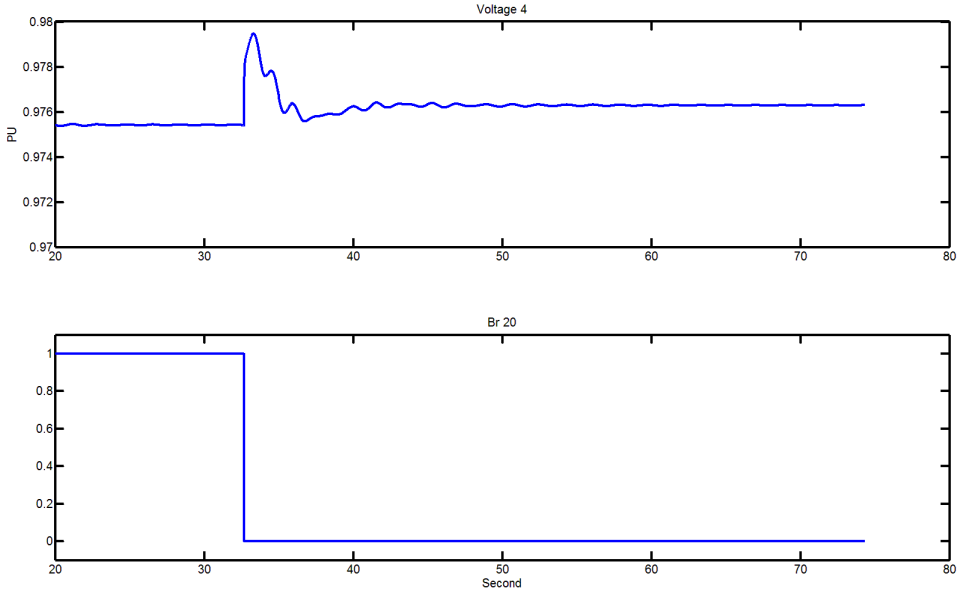


Fig. 4-14: Scenario 2

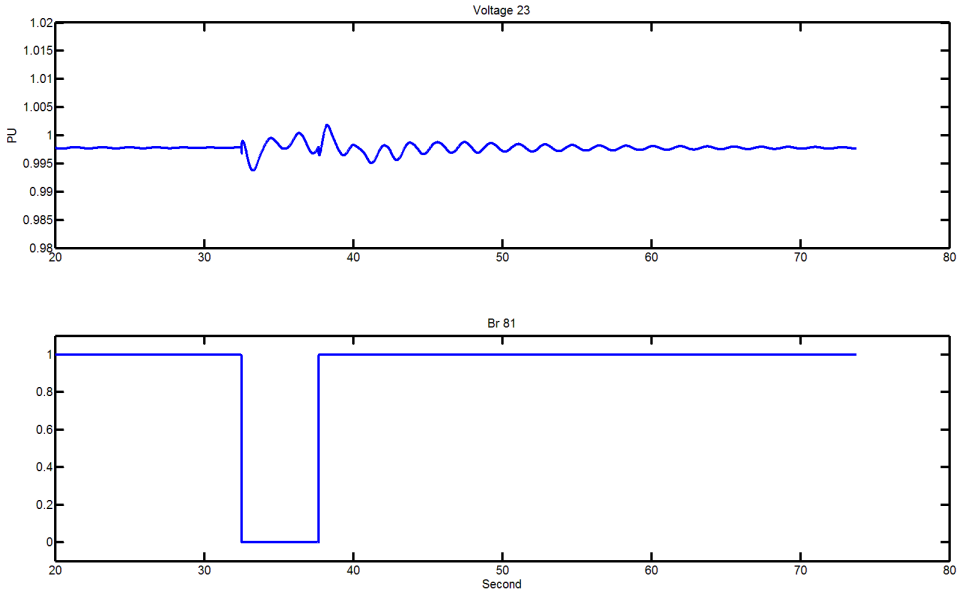


Fig. 4-15: Scenario 3

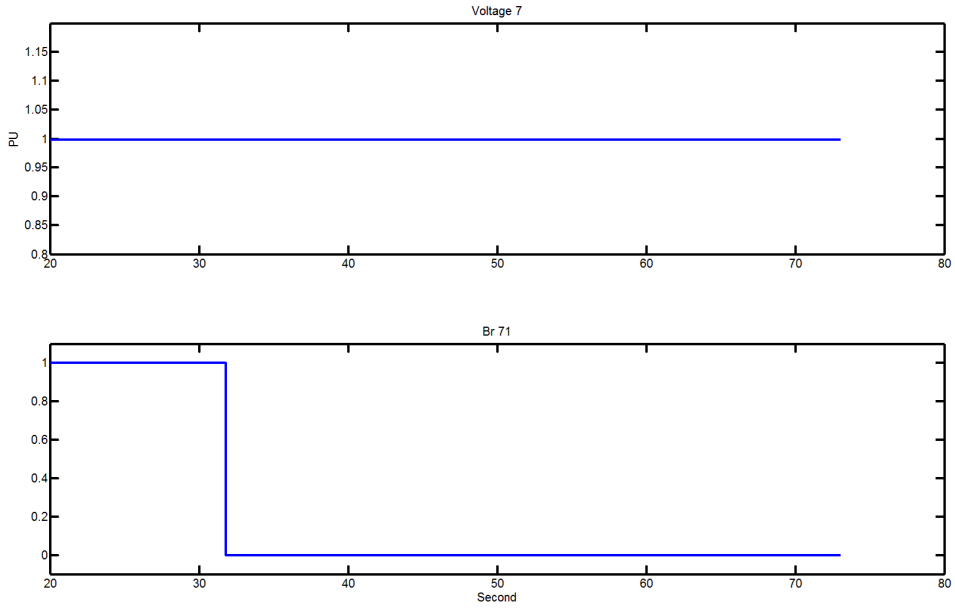


Fig. 4-16: Scenario 4

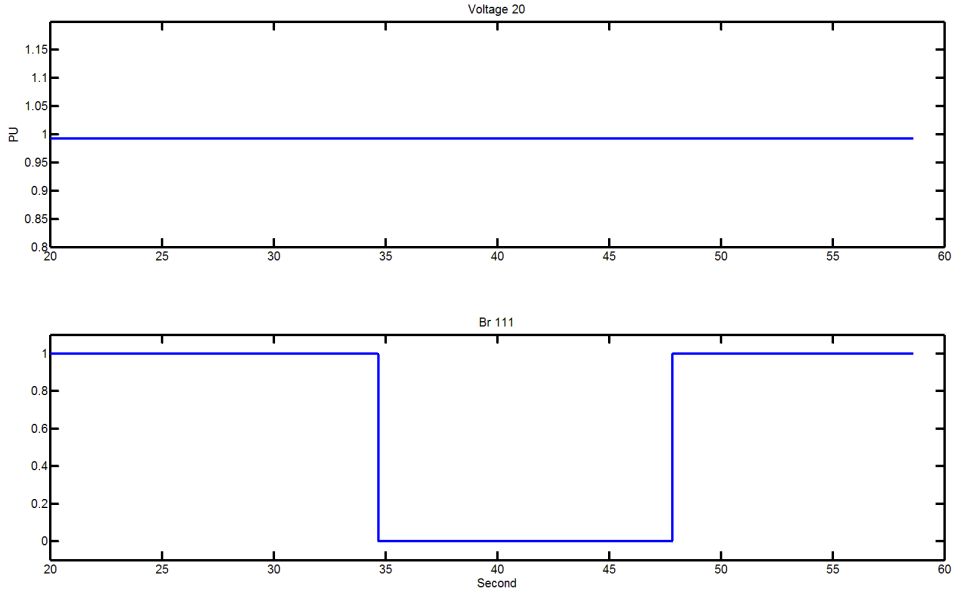


Fig. 4-17: Scenario 5

In both scenarios 4, and 5, a breaker within a substation is opened (substation 15 and 23 respectively), and that is the reason that no changes happen for the voltage waveform.

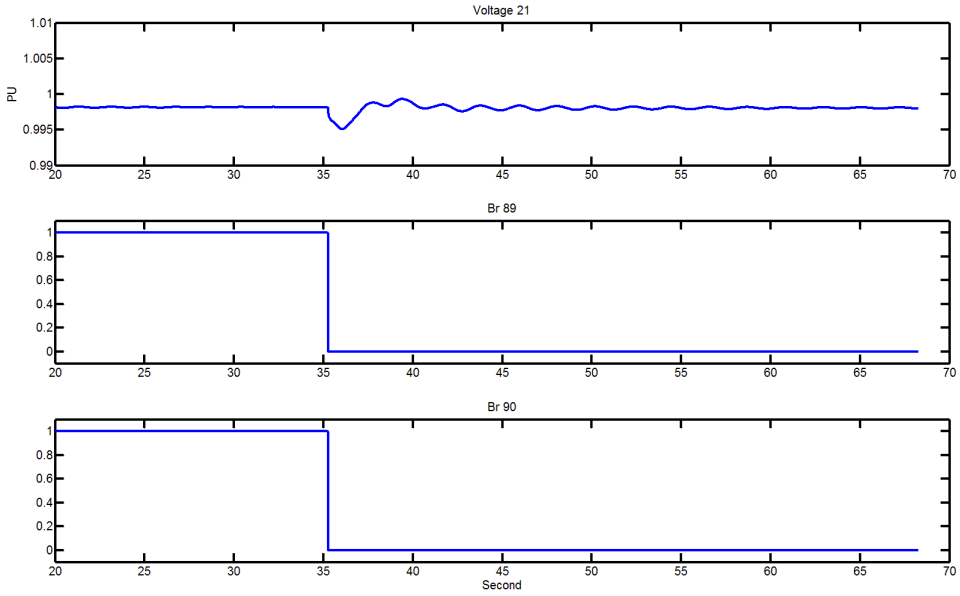


Fig. 4-18: Scenario 6

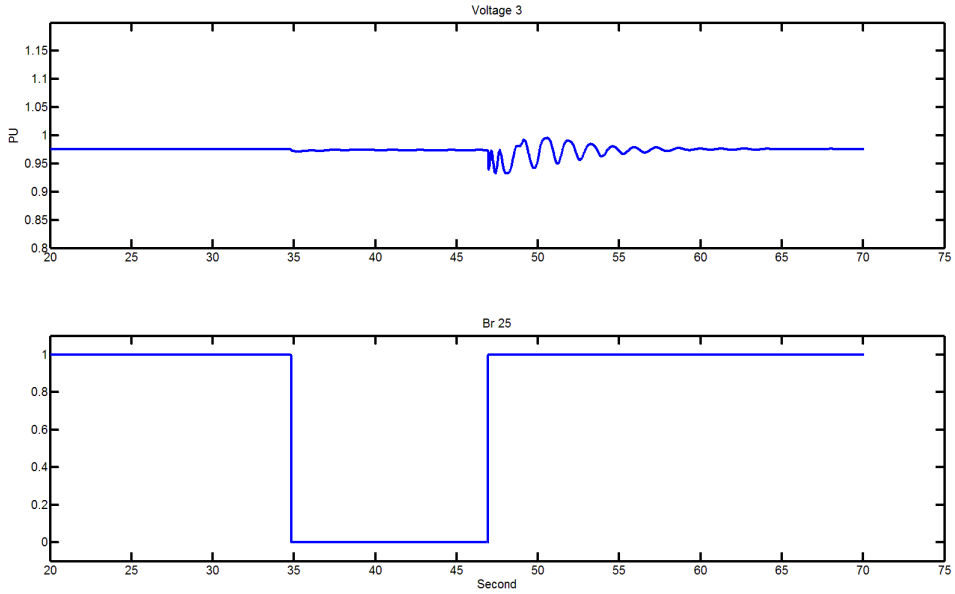


Fig. 4-19: Scenario 7

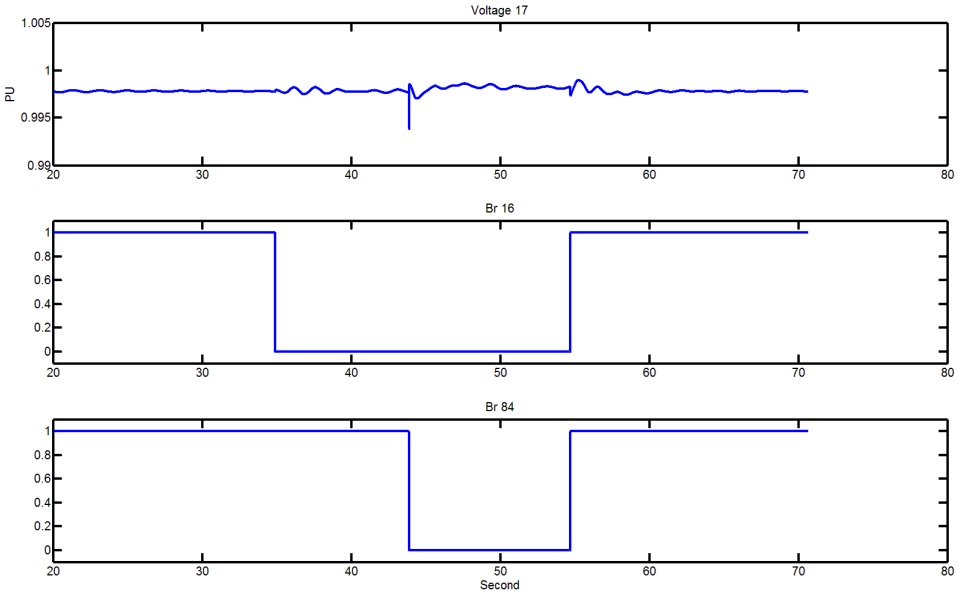


Fig. 4-20: Scenario 8

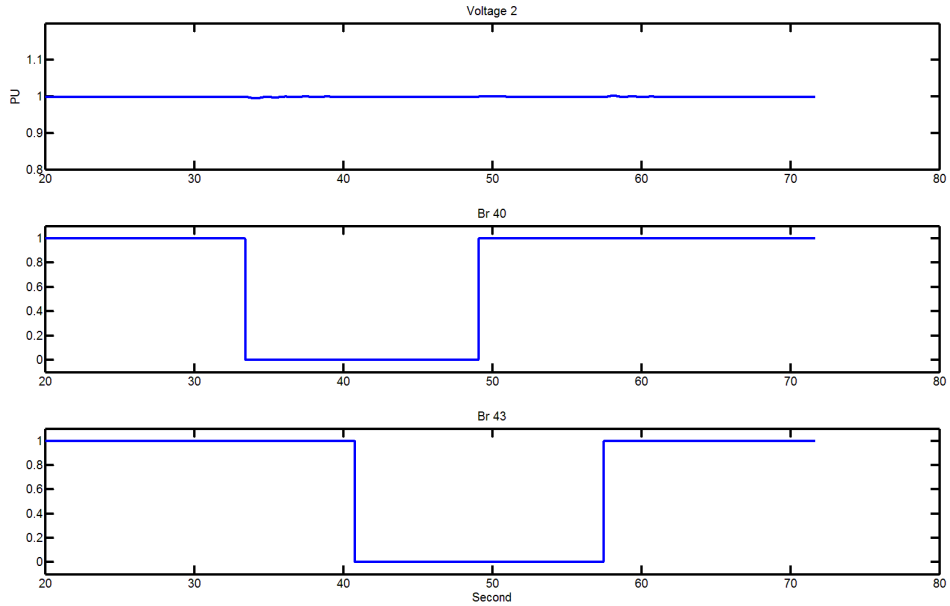


Fig. 4-21: Scenario 9

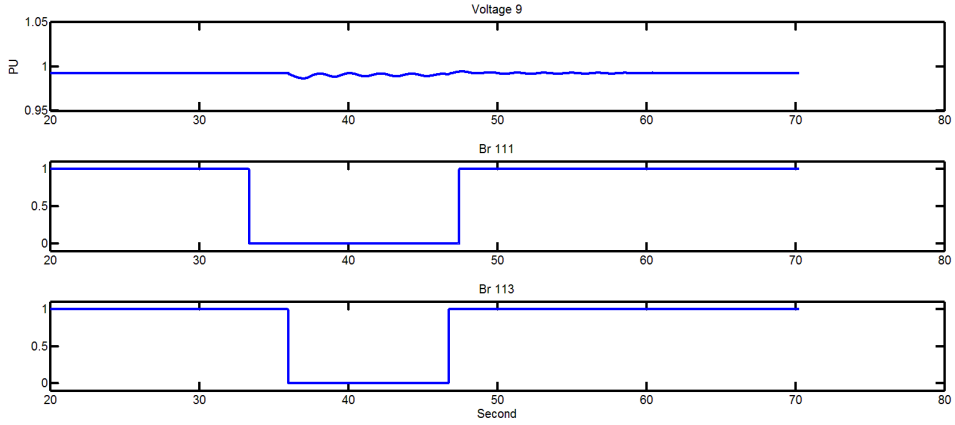


Fig. 4-22: Scenario 10

4.5. Chapter Summary

The chapter provides a brief overview of the modeling of the test system in SimPowerSystems (MATLAB/Simulink) and the simulation of the model in real time. As the detail modeling procedure and modifications are out of the scope of this thesis, the reader is referred to [25] to find more about how to implement a model in the Opal-RT real time simulator. The test system is explained in detail and the switching test scenarios are explained. Finally, some sample results for each scenario is shown. As the number of measured variables is 331, it is not possible to present all of the results in this thesis.

Chapter 5: Topology Processor, Algorithm and Coding

5.1. Introduction

In Chapter 3, the general steps involved in a typical topology processor were introduced and each of their respective procedures was explained descriptively. However, no specific algorithm or coding is detailed for these steps. This chapter scrutinizes previous work in topology processing, highlighting drawbacks and limitation of available topology processors, and to develop a new topology processor engine capable of overcoming current limitations, in addition to be capable of executing with conventional, PMU, or both data sources.

Firstly, previous works in this area are introduced. The approaches used for determining network topology are briefly discussed and deficiencies of each algorithm are highlighted. Next, proposed topology processing method in this thesis is presented and explained step by step. The algorithm is explained descriptively and illustrated using flowcharts in full detail. Relevant snippets of the MATLAB code of the algorithm are presented to reveal the intricacy of some implementation issues. This chapter provides sufficient information so that any reader with sufficient coding abilities is capable to implement the algorithm independently, and reproduce the results precisely.

5.2. Literature Review

Although topology processing is a crucial step necessary for most EMS applications to properly function (particularly state estimation), the openly available documented evidence on the industrial implementation of current topology processors and associated research shows a lack of rigorosity in the description of the available algorithms. In fact, such general explanations are insufficient for any practical implementation.

In the coming sections, the most relevant work in this area is scrutinized in detail. This gives a good background for the new algorithm proposed in this thesis and emphasizes the need for a practical implementation of a topology processor which is explained in detail so that it can be reproduced independently.

5.2. 1. Automatic Power System Network Topology Determination [8]

Introduced by A. M. Sasson et al, this is the cornerstone work in the area, and common reference for majority of other proposals. Similarly, the proposed method in this thesis builds on top of this algorithm. Therefore, a crucial discussion is provided next, highlighting existing defects and limitations in the algorithm.

The algorithm uses matrices: it reads the input data in the form of matrices and gives the output in the same form. The main idea is to detect substations in which some breaker change has occurred, and to start the analysis from those stations. The algorithm attempts to find whole the different closed paths within a substation, and to report all the components in each path as a unified station. Then if more than one closed path is found in each station, it means that the station is split and possible new stations have been formed. Similarly, it performs an analysis to determine if there are separate islands in the power network.

The whole algorithm consists of three modules. One performs offline data analysis, i.e. reads the input matrices with connectivity data and breaker initial statuses. The other is in charge of receiving telemetered breaker statuses and performing topology processing. The final module analyzes the availability of measurements and whether circuits are available for performing static analysis (power flow) or not.

Although this algorithm is the common reference in the area, it presents several drawbacks. The following are some of the major issues with this TP:

- The proposed algorithm is far from being practical as it is only descriptive. No flowchart or mathematical description is provided. This has rendered the method to be very obscure and limits practical implementation. The descriptions are highly intricate, creating more difficulties for its implementation.
  Just as one out of several examples, in one of the algorithm steps it is stated "*find a closed circuit breaker connected to the circuit; record in the same list the circuit on the other side of this circuit breaker. This circuit breaker need not be considered again…".* As it can be observed, the procedure is only described in words, and as this step is a part in a loop, it doesn't specifically express that how it should manipulate the loop in way not to consider each circuit breaker more than once per cycle! Another example is when the author wants to explain the islanding analysis procedure: "*when the condition arises that all the line and transformer circuits are available, but one or more substations have become separated, it is necessary to introduce phantom unavailable circuits that connect the separated parts of the substation",* but no information of how to introduce these phantom circuits (or how they should be connected to the other parts of the system) is provided. There are several other examples like this, that do not provide sufficient details for a full comprehension of the algorithm.

- Whenever a breaker status change happens, the algorithm needs to perform topology processing for the whole network again. It needs to check for breaker statuses which are different from the initial statuses, and process the topology for

all the substations with breakers statuses different from the initial ones. This is a considerable drawback.
First it decreases the speed of topology engine as it needs to determine the topologies even for those stations that have not changed since the previous cycle. Second, as the topology of these kinds of stations is re-determined with a change in some other parts of the system, the numbers assigned to the station or previously derived stations are changing from cycle to cycle, which is inefficient and difficult to trace.

- In addition, there are major concerns about the efficiency of the algorithm. Once the descriptive procedure is descripted and coded following the stated procedure, the performance will most likely be poor.
  First, the algorithm is just designed to deal with changes within a substation, and there is no consideration for changes of breakers which are not located in a substation configuration but are located on transmission lines. Second, the main assumption of the algorithm is that there is always an open circuit breaker in the system. The algorithm simply can't operate whenever there is no open circuit breaker in a substation. That means than whenever a change occurs in which all of the open circuit breakers within a substation are closed, the algorithm simply can't report the correct topology of that substation, and current topology of the whole system.

- Another important drawback of this TP is that it is unable to deal with the entire existing substation configuration in the real word. The test system which is used in the article only has the combination of "Breaker and a Half" and "Double Bus Double Breaker" configurations. The numbering rule which is introduced in the article does not work for configurations such as Ring. However the exact behavior of the algorithm dealing with different configuration is hard to predict as the algorithm is not rigorously documented.

- Next, the algorithm is not efficient when dealing with substation merging. What has been proposed in the article is just regarding the station splitting, and there no considerations for situations when some stations merge together. The algorithm processes the topology and compares it to a pre-defined initial topology; therefor, in the case that substation merging occurs, it may severely changes the numbering of those stations involved in the merging action. This will make the tracking of the changes almost impossible. In addition, this deficiency may even be worse when it comes to the predefined initial data structure. If some of the predefined initial stations have the possibility to merge together, this algorithm will inevitably fail to detect such merging scenarios.

- One of the crucial tasks of a typical topology engine is completely skipped and neglected in this TP. The TP has no ability to detect if any island is energized, de-energized or grounded. This is very important as those parts of the network which are not energized should not participate in state estimation, or static analysis (power flow solution).

- There is also a general problem with the automatic numbering of the network components. One of the tasks of a topology engine is to automatically assign numbers to splitting stations, or emerged islands. However, the descriptive explanation provides no information regarding on the numbering procedure. This limits the efficiency of any independent implementation.

- There are unnecessary redundancies in the primary data. The input matrices cover redundant information, which makes interpreting data unnecessarily complex. In addition, the algorithm itself requires the elaboration of lots of redundant lists to be used in sequential steps. A careful data preparation allows avoiding these unnecessarily listing.

The problems mentioned above are some of the major concerns with this TP which makes any independent implementation of such an algorithm impossible. One of the key characteristics for any algorithm to be useful and veracious is to be executable, i.e. to have the potential to be implemented and executed in reality. Otherwise, it is impossible to verify its accuracy, and/or results. Unfortunately, this method is far from the standards of an algorithm that can achieve any execution.

5.2. 2. Real-time Modeling of Power Networks [9]

Real-time modeling of power networks in [9] refers to a computer based representation of the power network which is based on mathematical equations. It also comprises different functions such as topology processor, observability analysis, state estimation, bad data detection and external modeling. In [9] all the required steps for Real-time modeling are explained and general procedures for each one are described. However, no specific algorithm or mathematical representation is provided to achieve this. This is also true for the topology processor proposed in [9]. The discussion on topology processor in this article is more or less a very short general explanation of a how a topology processor works, and what steps are necessary in its procedure. Therefore, this work does not offer any insight for practical implementations.

5.2. 3. A Topology Processor That Tracks Network Modifications Over Time [10]

This TP is completely based on the one discussed in 5.2.1. Actually, no specific algorithm is proposed for the topology processor itself; it only provides an algorithm to save the outputs of the topology engine for each execution cycle, and how to use this saved data in order to reduce the calculation burden in formulating the state estimation problem. The lack of any particular algorithm for the topology engine itself has made the verification of the proposed method impossible. Besides, as the extension is absolutely the same as one in 5.2.1 it is completely exposed to the same deficiencies as those discussed for that TP in [8]. Therefore, this work has no practical value as it does not provide any guidance on how to implement a real topology processor.

5.2. 4. A New Algorithm of Topology Analysis Based on PMU Information[12]

To the knowledge of the author, this work is the only one considering the use of PMUs for processing the topology. It propose a method completely different from those using matrices. Instead, it uses a current threshold through the lines. Although the idea is new, it is far from being practical.

First, the proposed algorithm can be used only when PMU data exists for currents through all the lines. Otherwise it can't be used. This seems to be an unlikely situation for most of the current power networks all over the world in any foreseeable future. Next, the algorithm is not rigorous. Many parts of a real practical topology processor are neglected; it does not address how to deal with different configurations, automatic numbering, islanding and energization checks.

The algorithm uses current thresholds to decide whether a line is connected or not. It also calculates the difference of the current flow in the present cycle and previous one, and if a great difference is shown, it concludes that a switch status has been changed. However, it is not completely clear how it deals with these changes and analyzes them. This is problematic when it comes to changes within a substation. The algorithm checks if the current Kirchhoff's law apply for a bus in a substation; if the law is not satisfied, it concludes that the substation is split. It provides no information regarding how to find the circuits belonging to those split or merged stations.

In addition, it does not state how to assign numbers to possibly new formed islands and substations. No any concrete example is provided, and the reader is just limited to some comments about general procedure. As a conclusion, the algorithm proposed is far from a practical application. The limitations are so high considering the mentioned drawbacks and probably it has zero efficiency in practice.

5.2. 5. A New Approach to Initializing and Updating The Topology of An Electrical Network [16]

The method introduced in [16] is completely different from what has been proposed previously as it is based on graph theory. The power system network is transformed to a graph, and then using a specific algorithm this graph is transformed into a sparse matrix. A mathematical procedure makes use of this sparse graph to solve for the network topology.

This method is probably faster than the previous ones, however there are some other concerns. First, the article documented in [16] is more of an industrious report rather than being a work upon which other approaches can emerge. The procedures in [16] are vague and fully developed to realize a TP in practice. Specifically, the algorithm which solves the sparse matrix to determine the network topology is just limited to a few number of mathematical equations and provides insufficient detail for an independent implementation.

Another major issue with this TP is that it is not clear how the algorithm is dealing with the changes inside the substations. Also, the procedure which transforms the power network to a graph is not clear considering that there are different substation

configurations. The reported results are just limited to how much time it takes to determine a network with certain amount of components.

Considering the mentioned defects, this work is very complex and obscure, making and independent implementation impossible.

5.3. Automated Topology Processing for Conventional, Phasor-Assisted and Phasor-Only State Estimators

As mentioned in Chapter 1 the aim of this thesis is to develop a robust network topology processor which can be further used in a PMU-only state estimator. On the other hand, the developed algorithm should have the capability of working both with traditional data and PMU data. In addition, section 5.2 completely revealed the need for a rigorous representation of a practical topology processor. This requires providing a fully developed algorithm than can deal with any kind of changes in the network either inside or outside of a substation. In this section the proposed algorithm in this thesis is fully explained and each of the procedural steps are covered in sufficient detail so as to make it possible for anyone with enough knowledge to build a practical topology processor based on the provided instructions in this thesis.

Moreover, evidence of an actual coded implementation is provided in the form of MATLAB snippets in this chapter, and rigorous testing in Chapter 6. Furthermore, once the results reported in this thesis are published, the implementation code may be distributed to interested parties in conditions to be specified.

5.3.1. Aspects of the new algorithm,

The new proposed algorithm is built on top of the one proposed in [8]. However, there are lots of new aspects proposed in this new algorithm such as:

1- A resolution of all the drawbacks discussed in section 5.2.
2- The ability to operate both with traditional data, PMU-only, or both.
3- It performs the topology processing just for those parts of the system that have suffered changes compared to the previous execution cycle, and not the initial pre-defined statuses. This makes the algorithm much faster.
4- It deals both with splitting and merging substations efficiently.
5- The algorithm is straightforward and easy to be implemented.
6- All the necessary parts of a practical topology engine are considered, including islanding and energization analysis.
7- The algorithm automatically assigns numbers to any recently formed island or splitting/merging of substations in an efficient way so that the changes in each part of the network can be quickly tracked.

These new aspects have made the proposed algorithm in this thesis robust and efficient to be implemented in practice. Rigorous tests are provided in Chapter 6 as evidence of these statements.

5.3.2. Basic rules

As the algorithm is built on top of that of [8], it shares some common rules. However, there are newly introduced rules. All of them are listed below:

1- Every substation should be assigned a number from *1* onwards.
2- Every breaker should be assigned a number from *1* onwards.
3- Every line should be assigned a number from *1* to *n*. Then, the buses in the system are assigned numbers from *1* onwards.
4- A breaker status is either 1 or 0. 1 represents the closed position, while 0 means that the breaker is open.
5- A breaker can't be connected to more than two circuits, i.e. lines or buses. In the case that there is a breaker connected to more than two circuits, phantom breakers should be introduced which are always closed.
6- For each line in the network, there is a static analysis (load flow) availability status. A line is listed as be available for the static analysis (load flow) if it is not open, and if it has available measurements in at least one of its ends.
7- Each breaker should be listed with one of the following four types:

   - Type 1: Breakers which are located on a generator line.
   - Type 2: Breakers which are located on a line connecting two different substations.
   - Type 3: Breakers which are located within a substation.
   - Type 4: Breakers which are located on a shunt line, i.e. loads or etc.

5.3.3. Inputs

The proposed algorithm in this method reads the input data in the form of two matrices.

- **Breaker Table Matrix**

This matrix contains information regarding breakers and circuits connected to them. If there are *N* breakers in the power network, the Breaker Table matrix would be a $N \times 8$ matrix. Each column holds specific information, as follows:

1- Breaker Numbers: The first column contains breakers numbers from *1* to *N*.
2- Near Substation Number: By near substation number, we refer to the closest substation to the breaker. For breaker types 1, 3 and 4 it is the substation which they belong to. For breakers type 2 which is located on lines which connect two different substations, the near one's number is entered here.
3- Far Substation Number: For breaker types 1, 3 and 4 it is the substation which they belong to. For breakers type 2 which is located on lines which connect two different substations, the further one's number is entered here.

4- Breaker Type: As said previously, each breaker should be listed as one of 4 available types.
5- Breaker status: It is either 1 or 0. 1 represents that the breaker is closed, while 0 means the breaker is open.
6- Circuit analysis: Based on the breaker's type, a breaker is either placed on a single circuit (1,2,4) or is within a substation and connected to two different circuits (3). For types 2, the entry in row 6 is the corresponding line number and the entry in row 7 will be 0. For types 3 the corresponding circuits' numbers are entered in rows 6 and 7. Please note that if this circuit is connected to more than one circuit breaker its number should be entered with a minus sign. Also, for types 1 and 4 the entries for both rows 6 and 7 are 0.
7- Refer to 6.

Notice:

The order of entries in rows 6 and 7 is not important for breakers of type 3.

8- Original Substation Number: For breakers of type 3, it is equal to number of substation to which they were originally part of when all the breakers are closed. For other types, the entry is zero.

- **Configuration Matrix**

The Configuration Matrix conveys information regarding different circuits in the power network. If there are *M* circuits in the network, i.e. *M* different lines and buses, the Configuration Matrix would be a $M \times 11$ matrix. Each column is dedicated to specific type of information:

1- Circuit Numbers: The first column contains circuit numbers from 1 to *M*.
2- Rows 2, 3, 4, 5, 6, and 8 should be explained together as the entries in these rows are somehow related to each other. A circuit can have different situations illustrated in Fig. 5-1 on the next page:

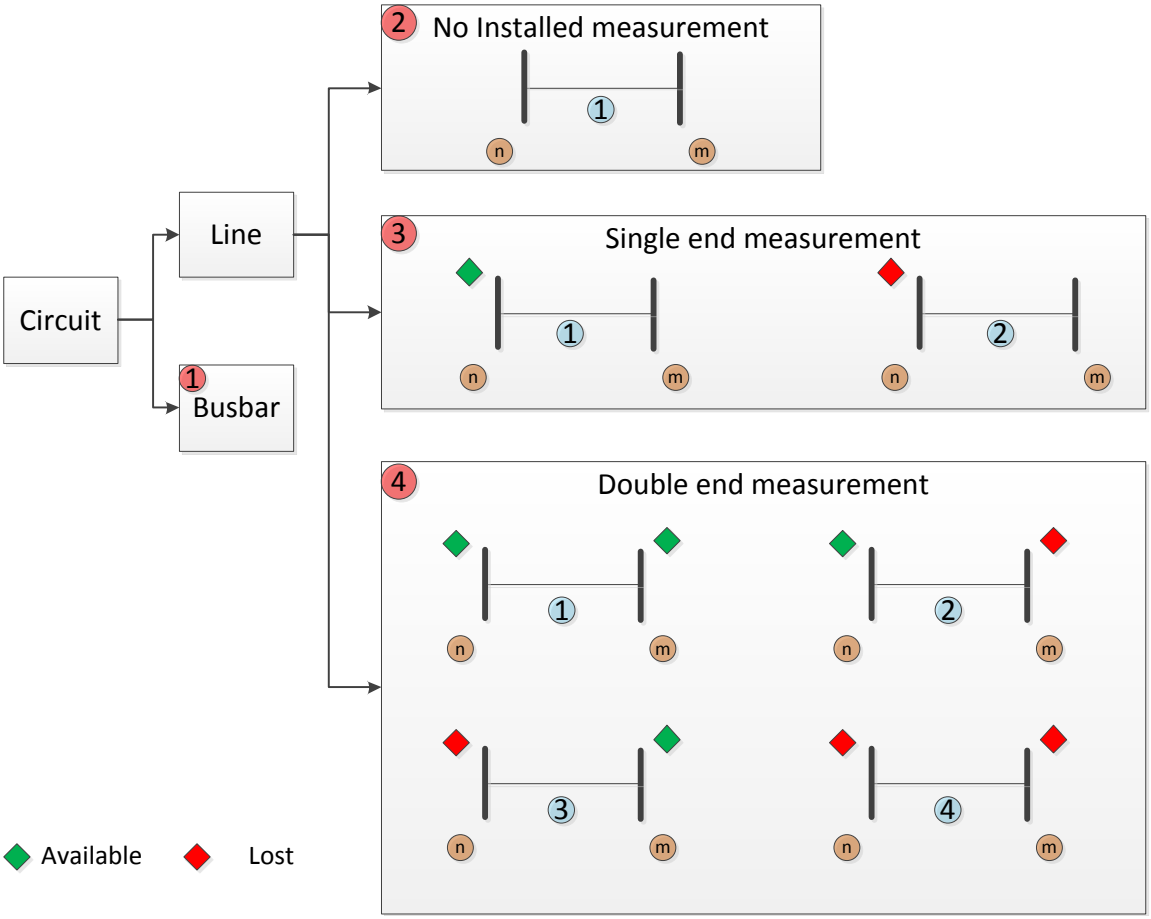


Fig. 5-1: Circuit Situation

For the case number 1, i.e. when the circuit is a bus, the number of the station to which the bus belongs, is entered in the second row, and all the rows 3, 4, 5, 6, and 8 entries are 0.

For case number 2, i.e. when the circuit is a line connecting stations n and m, regardless of the order, n and m are the entries of rows 2 and 3. All the rows 4, 5, 5, and 8 entries are 0.

For case 3.1, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is 0, the entry of row 5 is 1, the entry of row 6 is 0, and the entry of row 8 is 0.

For case 3.2, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is 0, the entry of row 5 is 0, the entry of row 6 is 0, and the entry of row 8 is 0.

For case 4.1, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is n, the entry of row 5 is 1, the entry of row 6 is 1, and the entry of row 8 is 1.

For case 4.2, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is n, the entry of row 5 is 1, the entry of row 6 is 0, and the entry of row 8 is 1.

For case 4.3, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is n, the entry of row 5 is 0, the entry of row 6 is 1, and the entry of row 8 is 1.

For case 4.4, the entry of row 2 is n, the entry of row 3 is m, the entry of row 4 is n, the entry of row 5 is 0, the entry of row 6 is 0, and the entry of row 8 is 1.

Notice:

Row 8 is just an informative column about whether a measurement is installed on both ends (1) and just on single end (0).

3- Refer to 2.
4- Refer to 2.
5- Refer to 2.
6- Refer to 2.
7- Circuit static analysis (Load Flow) Availability: It is 1 or 0. The entry of this column is obviously 0 if the circuit is a bus. In the case that the circuit is a line, 1 means that the circuit is available for load flow, i.e. it is not open and has at least one available measurement, and 0 means the line is not available for load flow.
8- Refer to 2.
9- Column 2 Substation Island: The number of the island to which the substation of the second column belongs.
10- Column 3 Substation Island: The number of the island to which the substation of third column belongs.
11- Column 2 Substation Original Number: The number assigned to the second column's substation initially when all the breakers are closed.
12- Column 3 Substation Original Number: The number assigned to the third column's substation initially when all the breakers are closed.

5.3.4. Outputs

The outputs of this algorithm are also in the form of different matrices.

- **Updated Breaker Table and Configuration Matrices**

Both the Breaker Table and Configuration Matrices which are explained in 5.3 are updated and ready to be used as an input for the next cycle. If any new substation has been split or any new islands have been formed, the number for which they have been assigned automatically is replaced with the previous number. If any circuit is open or disconnected a minus sign is introduced for its number in the configuration table.

- **Sub Matrix**

This matrix reports the substation in which any change has been occurred.

- **Substation Number Matrix**

The Substation Number Matrix holds information regarding splitting/merging stations. In fact, if there are L original substations in the system, i.e. when all the breakers are closed there are L substation, this matrix is a $L \times K$ matrix. The first column is the number assigned to original substations, and the other columns have numbers assigned to stations that are derived from this original substation. For example, if there are 4 stations in a system, and stations 1 and 3 each are split to 3 separate stations, the Substation Number Matrix can be:

1	5	6
2	0	0
3	7	8
4	0	0

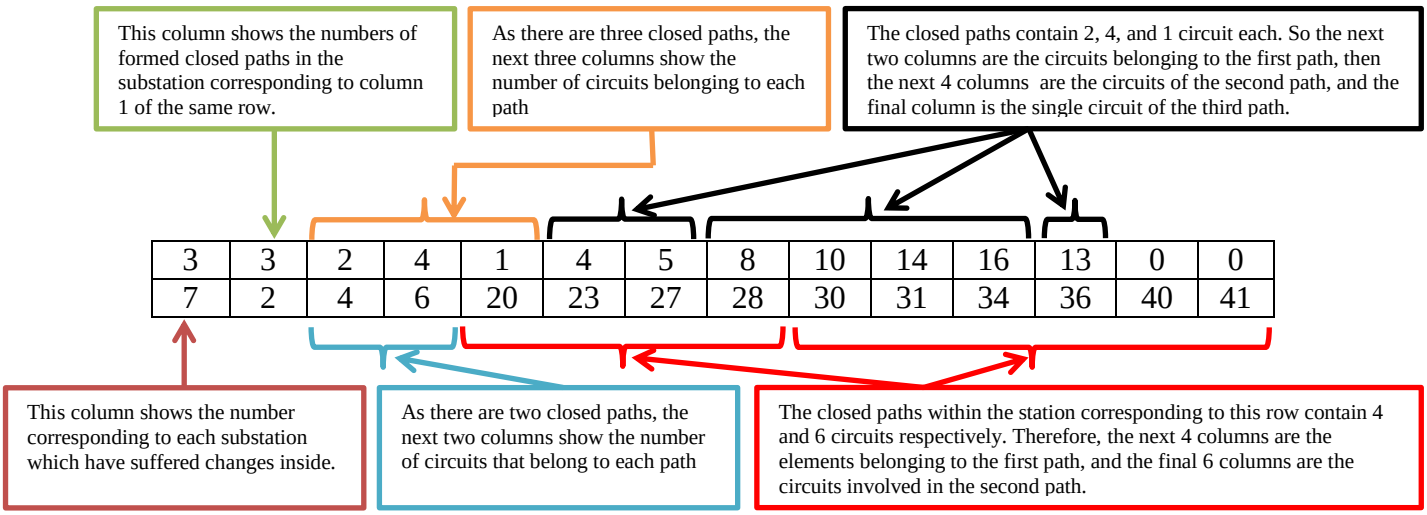
Tab. 5-1: Substation Number Matrix

- **List Matrix**

For each substation in which some changes have occurred, a list matrix is created. The number of rows is equal to the number of separate closed paths within a substation after a change has occurred. In the columns the circuits' numbers belonging to each close a path are entered. A closed path can contain just one single component (which means that circuit is disconnected) or consists of several circuits. In the case that a substation has two or more closed paths which consist of two or more circuits, that substation has been split.

- **Heart Matrix**

As the name suggest, it is one of the key output matrices. It is in a form that is very compact, and the same time contains most of the information necessary to interpret the network topology. As the matrix size is dependent on the changes that have been occurred in the system, it can be explained by an example. In a system with 10 substations, suppose that two substations have had changes in their breaker statuses, substation 3 and substation 7. Now assume that there are three different closed paths within substation number 3, while there are 2 separate closed paths in number 7. The circuits number that belong to each path in substation 3 are (4, 5), (8, 10, 14, 16), and (13); also the circuit numbers that belong to each path in substation 7 are (20, 23, 27, 28), and (30, 31, 34, 36, 40, 41). Now the Heart matrix for this cycle of changes is as shown in table 5-2 on next page:



Tab. 5-2: Heart Matrix

As it can be seen from the simple example above, the heart matrix contains valuable information in a compact form. This matrix helps in interpreting the topology changes in the system. For example, just by analyzing this matrix, the number of new formed substation, their corresponding circuits and the circuits that are disconnected due to changes within a substation can be identified and reported.

- **Island Matrix**

This matrix holds three different types of information. First the number of rows show the number of separate islands in the system. The numbers assigned to each islands is entered in the first column. Next, the columns are the substations which belong to the corresponding island of the same row. Finally, the last columns shows whether the islands is energized or not.

5.3.5 Automatic Topology Processing Algorithm[1]

The algorithm can be divided in six different parts as shown in Fig. 5-2 on next page:

[1] In this sections, some parts of the original code is used in order to make the explanations more tangible.

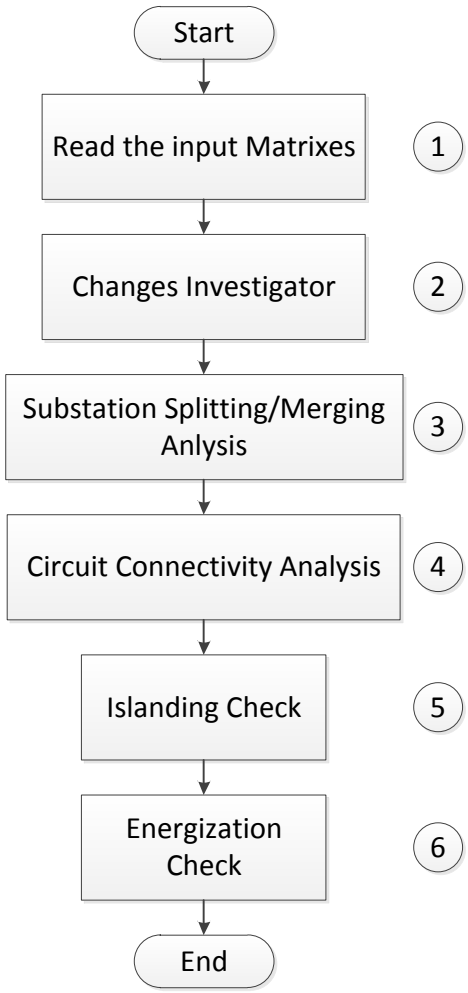


Fig. 5-2: Algorithm Flowchart

Each of these 6 sections are explained in sufficient detail below.

- **Read the Input Matrix**

This section is in charge of reading inputs, i.e. breakers' statuses.

- **Changes Investigator**

In this part the topology engine reads the input data and compares them with the data from previous execution cycle to see if any changes have happened. In the case that it detects some breaker status change, it scrutinizes the system to find out which breakers have had changes in their status. It saves both the found breakers' number and their corresponding original substation number in the case that the breaker is type 3. As the breakers of type 1 or 4 play no part in circuit connections, there is no need to save their changes. For breakers of type 2 the breaker number and its corresponding line should be saved also.

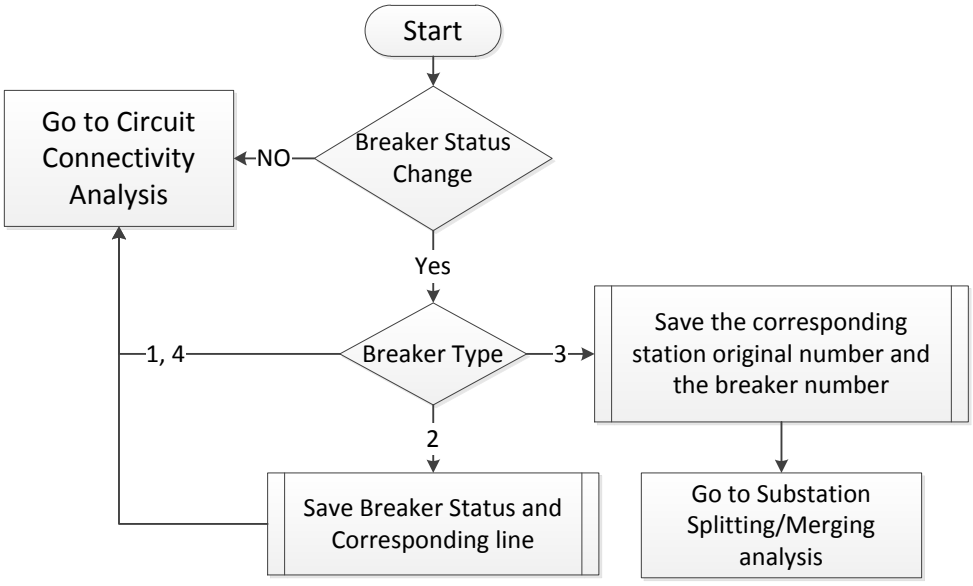


Fig. 5-3: Changes Investigator Flowchart

```
for i=1:number of breakers
    if Brn(i,5)~=Breaker_Table(i,5)
        if Breaker_Table(i,4)==3
            Out(a,1)=Breaker_Table(i,8);
            Out(a,2)=Breaker_Table(i,1);
            a=a+1;
        end
    end
end

for i=1:number of breakers
 if Brn(i,5)~=Breaker_Table(i,5)
     if Breaker_Table(i,4)==2
         Dis(1,di)=Breaker_Table(i,6);
         Dis(2,di)=Breaker_Table(i,5);
         di=di+1;
     end
 end
end
```

- **Substation Splitting/Merging Analysis**

If there are any breakers of type 3 with changes in the statuses, the program enters the section for scrutinizing changes within the substation. Otherwise it simply skips the section. This section investigates the original substations which have had changes inside, one by one. It is a very crucial step to scrutinize the original substation; e.g. assume that substation 1 has split in three substations 1, 8, and 10 sometime in the past. Now in the current execution cycle, the topology engine finds that there are some changes within substation 8; the engine starts to determine the topology for whole of substation 1 again. This makes the topology processor able of efficiently deal with the splitting/merging substations as it applies a kind of predictive method.

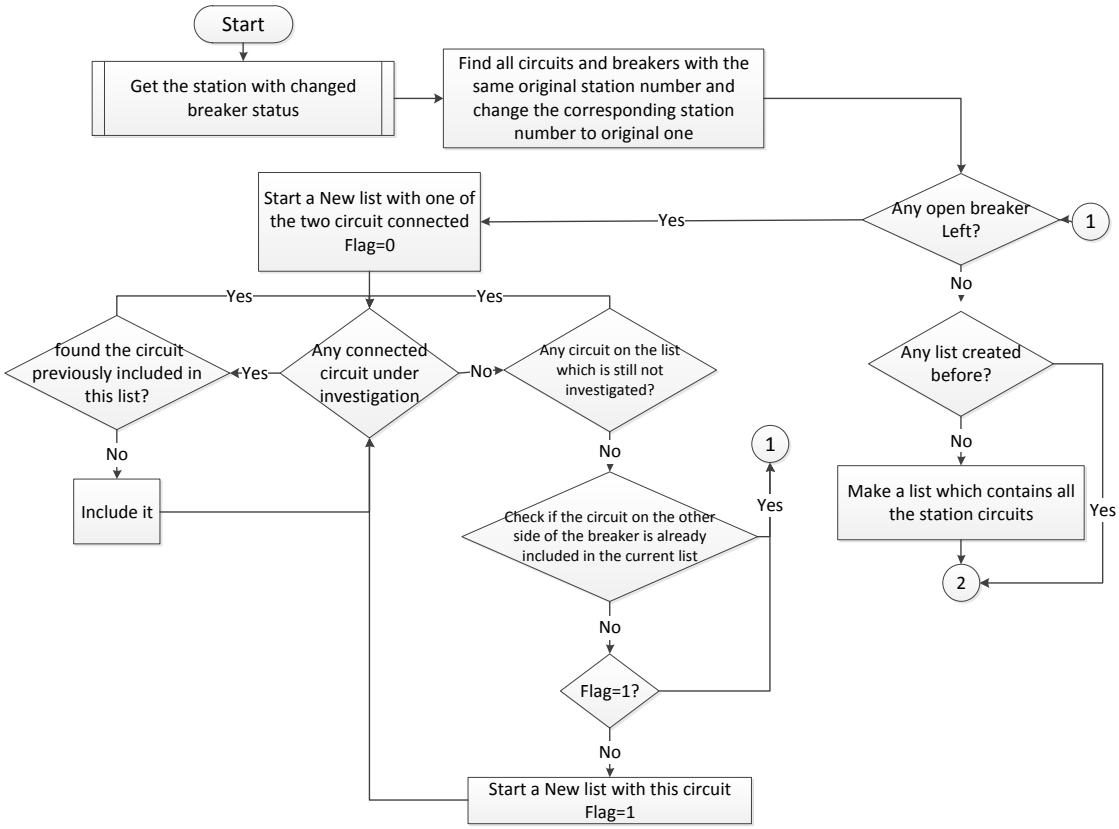


Fig. 5-4 (a): Substation Splitting/Merging

First the procedure in Fig. 5-4 (a) finds any open breaker within the station. If there are no open breakers, it simply makes one list for the station which consists all of the circuit breakers. Each list means a separate closed path, so when all the breakers are closed there is just one closed path within the station that contains all the station circuits. If it finds any open breakers, first it saves the breaker number. Then it starts a list with one of the two circuits connected to this open breaker. And by simply searching the Breaker Table Matrix, it finds all the other circuits which are connected through the closed circuit breakers to the one in the list. It saves all the circuit breaker numbers in a separate matrix, and adds the connected circuit which has been found to the list started before. Then it starts the same procedure for all the added circuits to the list one by one, until no any further circuits can be added. This procedure is done for all the open circuit breakers, until all the closed paths can be found.

```
i=1;while i<=f
      if List(d,f,p)<0
         for j=1:number of breakers
            if Breaker_Table(j,4)==3&&Breaker_Table(j,8)==Sub(1,p)
                if Breaker_Table(j,6)==List(d,i,p)&&Breaker_Table(j,5)==1
                    ⋮
                    f=f+1;
                    List(d,f,p)=Breaker_Table(j,7);
                    ⋮
                end
                if Breaker_Table(j,7)==List(d,i,p)&&Breaker_Table(j,5)==1
                    ⋮
```

```
                    f=f+1;
                    List(d,f,p)=Breaker_Table(j,6);
                    ⋮
                end
            end
        end
    end
i=i+1;
end
```

Please note that before any new entry is added to the list, a redundancy check should be performed to block redundant circuit numbers to enter in the same list. The procedure checks if the breaker under investigations has been previously entered in the used breakers list, or if the circuits that are going to be added previously exist in the list.
The output of this section is the List Matrix. Whenever the List Matrix is created the existing lists are analyzed to see how many components each one contains, and what those components' numbers are. The output of this analysis is the Heart Matrix.

```
for i=1:number of stations with changes inside
    in=0;
    hmatrix(i,1)=Sub Matrix(1,i);
    for j=1:number of stations with changes inside
        if List(j,1,i)~=0
            in=in+1;
        end
    end
    hmatrix(i,2)=in;
    for k=1:in
        ⋮
        hmatrix(i,k+in+j+counter+1)=abs(List(k,j,i));
        ⋮
    end
    counter=0;
end
```

The next step is to assign numbers to new possibly formed substations. For each substation, if there are two or more lists which contain more than one circuit it means that the original substation with changes inside has been split. To assign numbers to these new formed stations, the first step is to build a vector of unoccupied numbers. This vector is named *Possibility Vector*. The Possibility Vector is a vector from 1 to *m*, i.e. a number which is greater than the total number of original stations by a safe margin (usually 5 times greater). After the substation with changes has been recognized, from the *Substation Number Matrix* of the previous cycle, the numbers assigned to previously split stations are replaced by a 0 in the Possibility Vector (which means this number can be assigned in further steps).

```
for i=1:a number 5 times greater than the number of original stations
    Pvector(1,i)=i;
end
for i=1:number of original substations
    quit=0;
    for j=1:number of Sub Matrix columns
```

```
        :
        if SB(i,j)>24
            int1=SB(i,j);
            max(1,int1)=0;
        end
    end
end
```

After the Possibility Vector is formed, the new formed stations are assigned a number from the first possible number onwards. By number assignation, we refer to two different tasks. The first task is to find all the circuits and breakers belonging to the new substation, and replace their previous corresponding substation with a new one, and second is to build the *Substation Number Matrix* to be used in the next cycle.

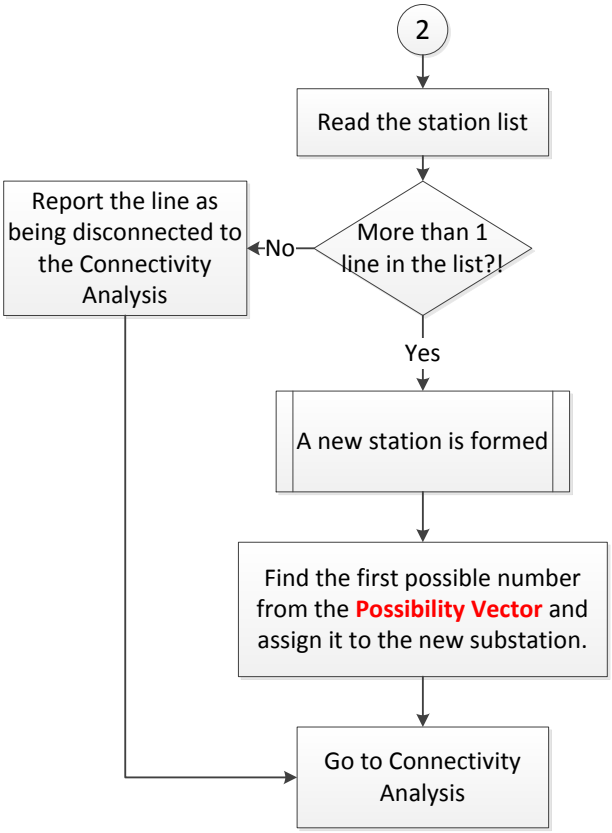


Fig. 5-4 (b): Substation Splitting/Merging

```
Max1=24;
for i=1:number of stations with changes inside
    in=hmatrix(i,2);
    counter=0;
    for j=1:in
        in1=0;
        :
        in2=hmatrix(i,in+in1+2);
        :
        max=max+1;
        :
        Configuration(in2,2)=Pvector(1,max);
           for br=1:number of breakers
```

```
            if abs(Breaker_Table(br,6))==Configuration(in2,1)
              if Breaker_Table(br,8)==Configuration(in2,11)
                  Breaker_Table(br,2)=Pvector(1,max);
              end
            end
        `    :
           end
           :
            counter=counter+1;
            if counter>1
            for m=1:j-1
                in1=in1+hmatrix(i,m+2);
            end
            max=max+1;
            :
            in3=hmatrix(i,j+2);
            for q=1:in3
                in2=hmatrix(i,in+in1+q+2);
                if hmatrix(i,1)==Con(in2,2)
                  Con(in2,2)=max(1,max1);
                    for br=1:number of breakers
                      if abs(Breaker_Table(br,6))==Con(in2,1)
                        if Breaker_Table(br,8)==Con(in2,11)
                            Breaker_Table(br,2)=max(1,max1);
                        end
                       end
                       :
                      end
:
end
```

The next step is to build the *Substation Number Matrix*. This is done by a simple search in the Breaker Table Matrix and find out which substation number is entered for breakers of type 3 which belong to the original substation with changes inside.

```
SB=0;
for i=1:number of original substations
    co=1;
    SB(i,1)=i;
    for j=1:number of breakers
        :
        if Breaker_Number(j,2)~=Breaker_Number(j,3)
           if Breaker_Number(j,8)==i && Breaker_Number(j,4)==3
            if Breaker_Number(j,2)>24
               :
                co=co+1;
                SB(i,co)=Breaker_Number(j,2);
              :.
                co=co+1;
                SB(i,co)=Breaker_Number(j,3);
             end
            end
        end
     end
   end
```

- **Circuit Connectivity Analysis**

This analysis is performed in two different steps within the algorithm. One step is performed simultaneously during the splitting merging analysis as shown in Fig. 5-4(b). The splitting/merging analysis is scrutinizing the *Heart Matrix* to check the lists; in the case that there is just one circuit in the list under investigation, it simply reports that circuit as one being disconnected.
The other part is dedicated to breakers of type 2. Actually, if the breaker status is 0, the topology engine finds the corresponding line by a simple search in the Breaker Table Matrix, and enters its number to list of disconnected lines. If the breaker status of type 2 has changed from 0 to 1, the sections search for all the other breakers of type 2 corresponding to the same line. If all of them are closed, the section reports that the line has transitioned from disconnected to connected.

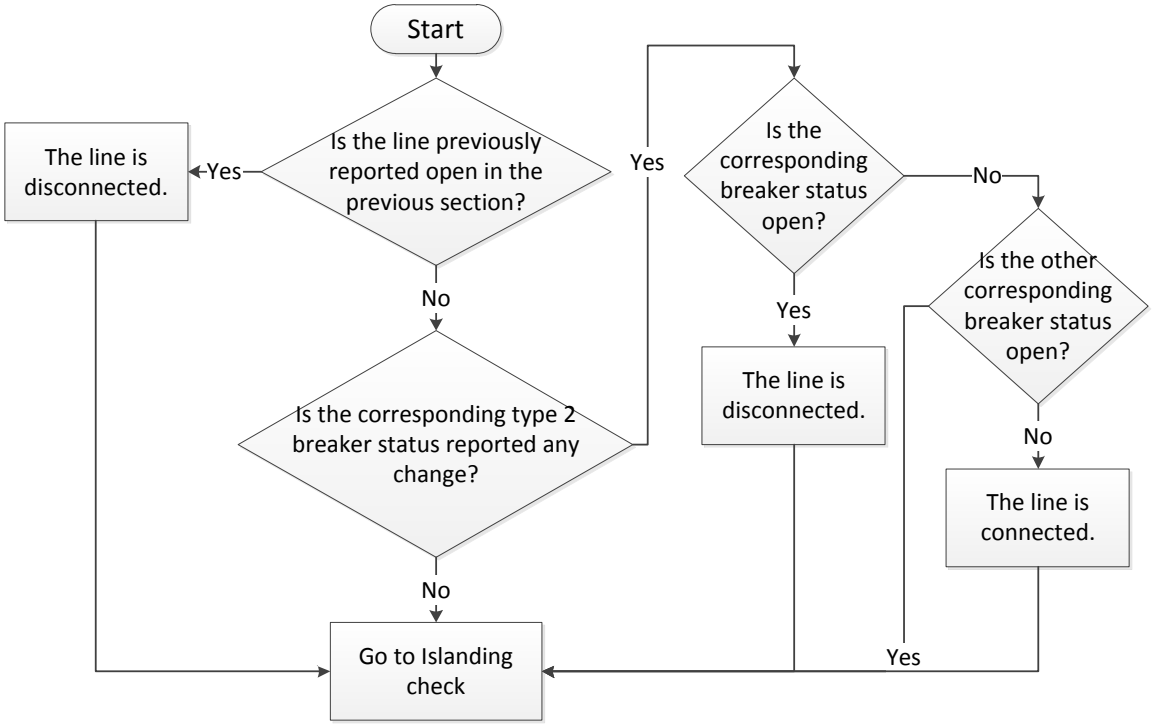


Fig. 5-5: Circuit Connectivity

```
for i=1:number Dis Matrix columns[1]
 op=0;
 op1=Dis(1,i);
 if Dis(2,i)==0 && Con(op1,2)>0
     Con(op1,2)=-Con(op1,2);
 ⋮
 end
end
```

- **Islanding Check**

[1] Refer to Change Investigator section.

The approach of this step is very similar to the one used when checking for changes within the substations. A list starts with the first substation and then the entire mutually connected substations are added. The same analysis is performed on all the added stations and at the same time a redundancy check is performed so that there is no additional entry of the same station number in the same list twice. If more than one list is created that means there is more than one island in the network.

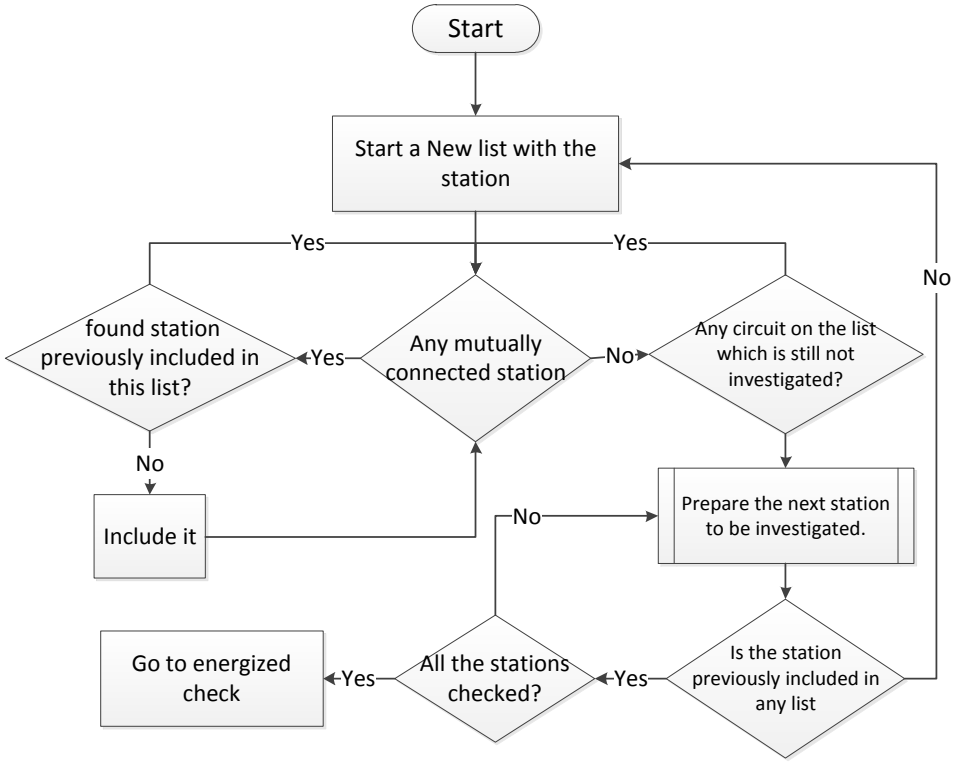


Fig. 5-6: Islanding Check

```
for i=1:max1
 ⋮
 if flag==0 %This is if for redundancy check
     f=1;
     d=d+1;
     List1(d,f)=i;
     t=1;
     while t<=f
        for p=1:number of lines
            flag1=0;
             if Con(p,2)==List1(d,t)
                  ⋮
                   f=f+1;
                   List1(d,f)=Con(p,3);
             end
             if Con(p,3)==List1(d,t)
                  ⋮
                  f=f+1;
                  List1(d,f)=Con(p,2);
              end
          end
```

```
        end
        t=t+1;
 end
end
```

- **Energization Check**

The energization check for each island is performed by searching for any closed breaker of type 4 connected to one of the stations belonging to that island.

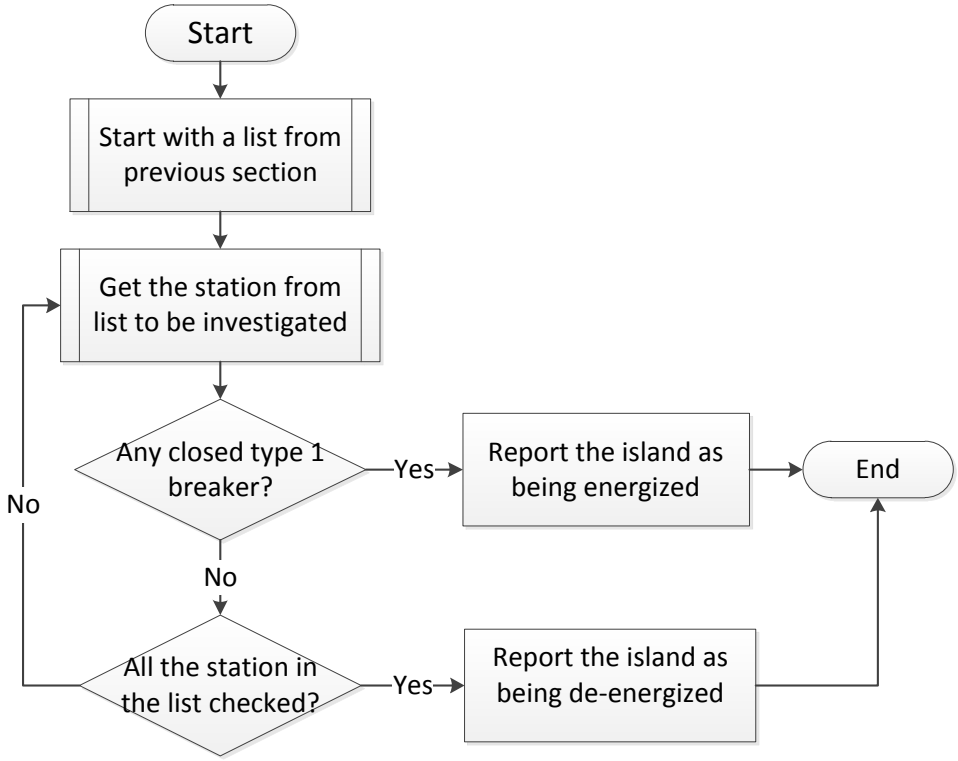


Fig. 5-7: Energized Check

```
for i=1:number of islands
 flag=0;
 for j=1:number of Island Matrix columns
   for k=1:number of breakers
     if Breaker_Table(k,4)==1 && Breaker_Table(k,5)==1
       if Breaker_Table(k,2)==List1(i,j)
             flagd=1;
             break
       end
       ⋮
       if flagd==1
         Check(i)=1;
       else
         Check(i)=0
       end
       ⋮
```

```
end
```

5.3.5 PMU Extension

As mentioned before, the proposed algorithm in this thesis is also able to work with data from PMU. Previously, there is one TP claiming to work just with PMU data [12]. The deficiencies of this TP are discussed in section 5.2.4. A crucial problem with the method in [12] is that it can only work with PMU data and can't work with traditional data. However, the algorithm in this thesis is robust enough to work only with traditional data, or to use PMU data as a support to verify the traditional data, or to work just using PMU data. Actually, as the algorithm works perfectly with traditional data, the inclusion of PMU measurements is also possible. If an algorithm only works with PMU data, the application is not reversible, and it may not work with traditional data properly.

To use PMU measurements, two different types of data can be used. First, the digital data which directly gives the status of the system breakers which is included in the IEEE C37.118-2011 [40] synchrophasor data. Second, the current magnitude through the lines can also be exploited.

- Working in PMU-assisted execution:

  By PMU-assisted mode, we refer to an execution condition which the topology engine uses PMU data to verify the breaker status changes in the system. This can be done in three different ways:

  - To check PMU digital measurements for the breaker that shows changes in its status.
  - To check the current flow magnitude through the line corresponds to the breaker which shows a status change. If the absolute value of its difference with the same current magnitude of previous cycle *(k)* is more than 100 times greater than the difference of instance *(k)* and *(k-1)*, verifies that a breaker status change has occurred. If the breaker is located within the substation, check for one of the buses current:

$$\left|I^{(k+1)} - I^{k}\right| > 100 \times \left|I^{k} - I^{(k-1)}\right| \qquad (5.1)$$

  - To check the current flow phase through the line that corresponds to the breaker which shows a status change. If the breaker is located within the substation, check for one of the buses current. This check is performed as shown below:

$$\left|\Phi^{(k+1)} - \Phi^{k}\right| > 100 \times \left|\Phi^{k} - \Phi^{(k-1)}\right| \qquad (5.2)$$

The three verifications methods which are mentioned above can be also used together; this is needed whenever the verification is of high importance and should have reliability near to 100%.

Notice:

In both equations (5.1) and (5.2), the switching occurs at instant *(k+1)*.

- In addition, the occurrence of any change within the substation can be verified by checking the current or phase of one of the buses that belongs to the station. This is the same approach as the one introduced in [12]. After the topology engine gives its results, if any splitting/merging is reported for the substations with changes inside, the final results can be verified (if needed) by checking if Kirchhoff's current rule applies for all the buses that belong to each station. This verification is not very reliable, though if Kirchhoff's current law is not satisfied, further investigation is needed. By satisfying of Kirchhoff's current law we refer to the following equation [12]:

$$\left|\dot{I}_a + \dot{I}_b + \cdots + \dot{I}_n\right| < n \times 0.2\% \times \max\left\{\left|\dot{I}_a\right|, \left|\dot{I}_b\right|, \cdots, \left|\dot{I}_n\right|\right\} \qquad (5.3)$$

Here, I_a, I_b,… and I_n are all the current that is injected to the bus with the same direction.

- PMU data only execution:

Whenever the topology engine is going to be used only with PMU data, it needs to use the digital data regarding the breakers status, and no major modification is needed. The verification methods mentioned above can be used here too. For example, all the three mentioned methods can be used to verify the received digital data regarding breakers statuses. In addition, a combination of these approaches can also be applied to increase reliability.

For any breaker of types 1, 2, and 4 for which digital data is lost temporarily, the corresponding line current magnitude can be used as an indication of whether the switch is closed or not. Ideally, whenever a breaker is open no current is passing through its corresponding line. However this is not the case in reality, as the open line still shows some current flow. In order to deal with this phenomenon, the same approach of [12] is used, i.e. to introduce a current threshold. If the current magnitude is below the threshold, the line is considered to be open. The threshold is considered to be equal to 0.08 p.u. [12]:

$$I > 0.08PU \quad \text{Line is connected}$$

But if the current was below the threshold, additional check should be done in order to find the line status. Suppose that the line connects substation n to substation m:

$$\left|\left|V_m\right| - \left|V_n\right|\right| > 0.01PU \quad \text{Line is disconnected}$$

The difference between the voltage magnitudes of two substations is calculated; if it was above the voltage threshold, the line is disconnected. Otherwise it is connected.

Notice:

Reference [12] was scrutinized in the literature review, and their drawbacks were stated. The reader may ask then that why this reference is cited in two parts of this section. One part is the splitting/merging substation verification. Although the approach the same as [12] is used, the reader should be aware that there are basic differences. In this thesis, the Kirchhoff's law approach is used only to "verify" that the results of topology engine. That means that once the TP reported the splitting or merging statins, the approach is used to quickly check the results. However, in [12] it is used to define the splitting/merging stations itself. It provides no information regarding any algorithm on how to choose the stations, how to assign numbers, how to find the corresponding circuits, etc. The approach is a suitable one for verification, but is very deficient to be used as the main method for splitting/merging analysis. Also, no any simulated waveform or real results are provided in [12] to support the author's claims.

Another part that [12] is cited in this section is the line status determination using current flow threshold. First, it is just an acceptable approach for temporary usage when no other data is available. The reason is that when a line is disconnected, there may be other mutually connected lines that may be affected heavily. Further meticulous investigation is needed for each power network, as disconnection of a line may results in fake disconnection of other lines using current threshold approach. An example of such a situation is provided in Fig. 5-8 on the next page. As it can be seen in that figure, breaker 117 is opened so line 42 is disconnected. As a result, the current in line 42 is decreased suddenly to 0.0002648 p.u. However, this has caused the current in line 12 also to be almost equal to zero (below the 0.08 p.u. threshold). To see where these lines and breaker are located in the system, refer to Fig. 6-2 on page 72.

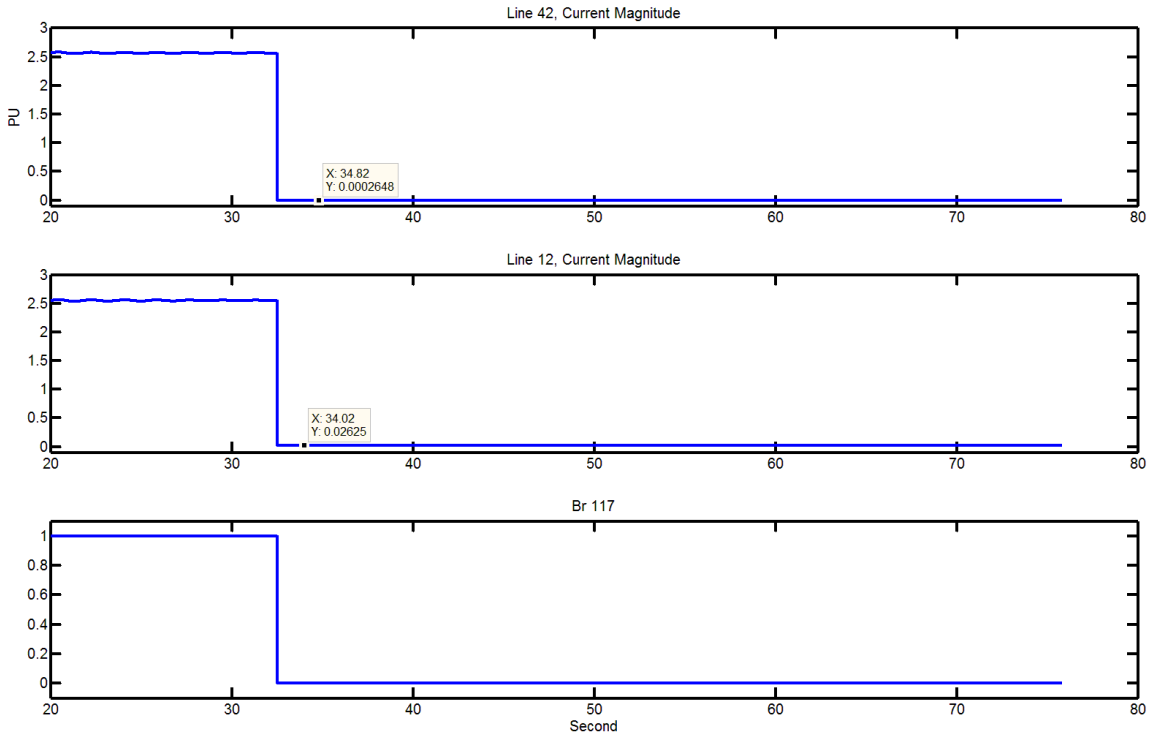


Fig. 5-8: Current Flow Magnitudes

For the work in [12], it's a major problem as this is the main method to define line statuses. However, this is not an issue for the algorithm in this thesis, because it uses the threshold method as a backup. In addition, the strategy of the proposed algorithm in this thesis is in a way that it avoids such an error by additional checking of station voltages. Here line 42 which is the genuine disconnected line, is the link between stations 15 and 24. Line 12 which is the line affected by line 42 disconnection, is a link between stations 3 and 24. The voltages are shown in Fig. 5-9 and Fig. 5-10 on next page:

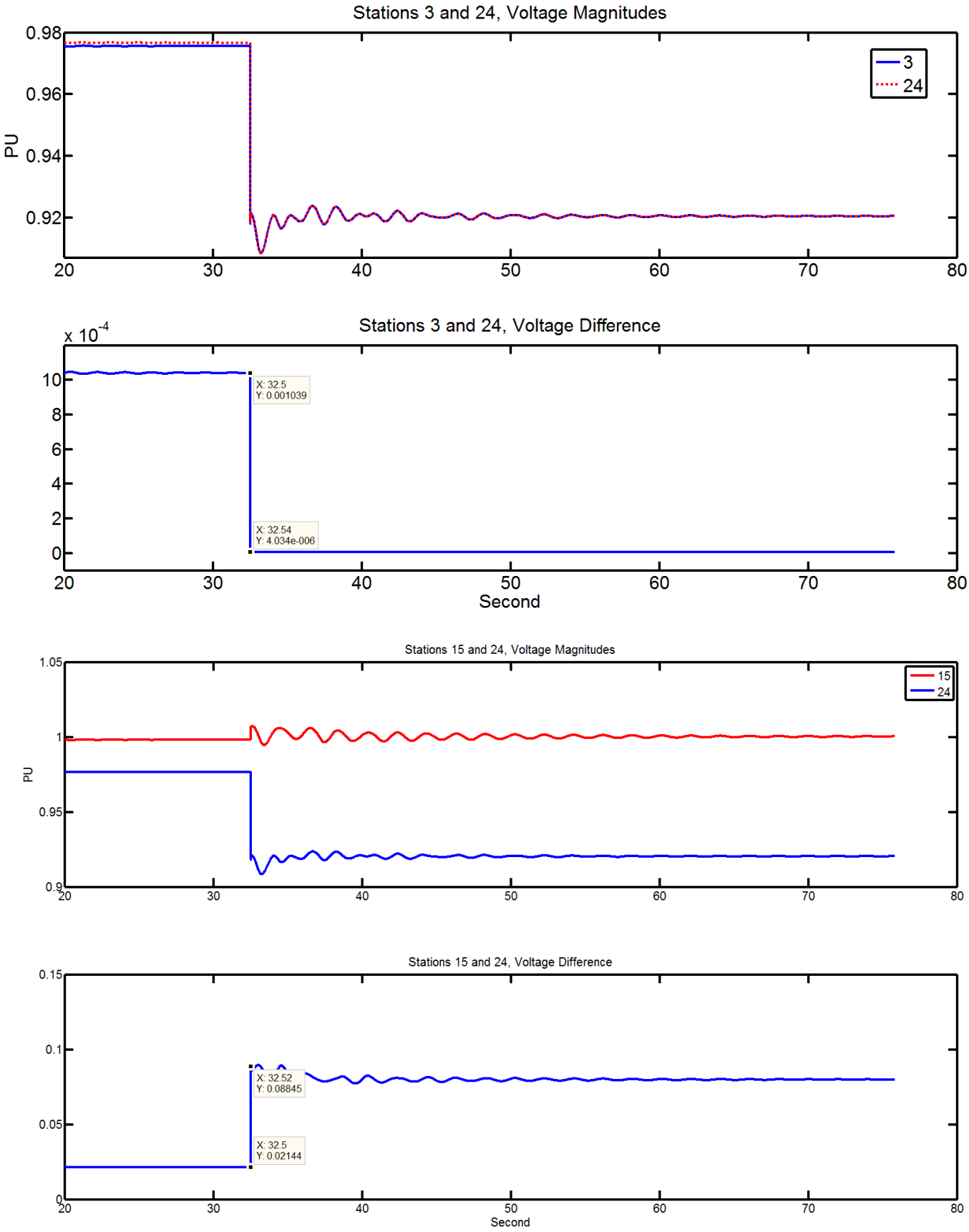


Fig. 5-9 and Fig. 5-10: Stations Voltages

As it can be observed in Fig. 5-9 and Fig. 5-10, the voltage difference before switching is almost *0.001p.u.* for stations 3 and 24, and is almost *0.021p.u.* for stations 24 and 15. However, after the switching, the voltage difference is almost 4×10^{-6}*p.u.* for stations 3 and 24 (remains below threshold); for stations 15 and 24, it reaches almost *0.09p.u.* which is far above the threshold of *0.01p.u.* and shows that the link between stations 15 and 24 is really disconnected.
The reason for this approach is that when a line has a flow below *0.08p.u.*but is not practically open, the voltage loss through the line is very low; this results in a very low voltage difference between the stations that are connected by that line.

5.4. Chapter Summary

In this chapter, previous work in the area of topology processing was discussed, and current drawbacks and limitations were identified. The need for a robust topology processor that is able to deal with all the different aspects needed for a topology engine was emphasized by showing how the available topology processing approaches have limitations. Then new using algorithm was presented both in a descriptive manner and by using flowcharts. Finally it was discussed how the proposed method can work using PMU data in assistance to traditional data, or just work using PMU data.

Chapter 6: ATP Testing

6.1. Introduction

This chapter provides a series of rigorous tests on the new topology processing algorithm, as evidence of its implementation, robustness and efficiency. The topology engine is tested over two different test systems. First, these two different test systems are introduced and enough information is provided regarding their respective topological characteristics. Then, the performed chained-scenarios are introduced. By chained-scenarios we refer to series of topology changes which occur from cycle to cycle. Finally, the outputs of the topology engine for each scenario are also explained.

6.2. Test Systems Overview

There are two different test systems used in this thesis. One is the same as the one which is used in reference [8], and the other one is IEEE Reliability Test System 1996 [14]. The topological characteristics of each system are discussed briefly in this section.

6.2.1. Conceptual Test System [8]

The overview of this test system is shown in Fig. 6-1:

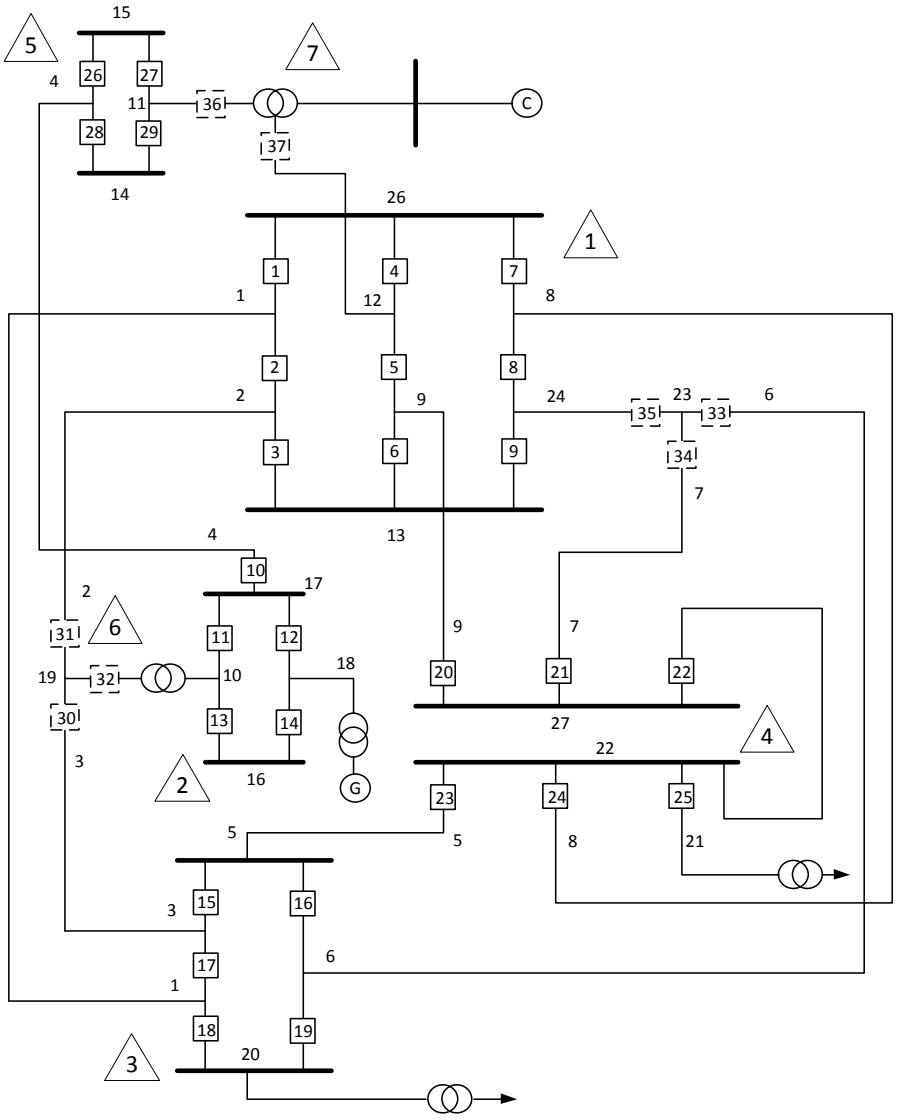


Fig. 6-1: Conceptual Test System

As it can be seen in Fig. 6-2, this system consists of 7 different substations. In addition, there are 37 circuit breakers and 27 circuits in the system. The breakers which are shown by dashed lines are the phantom circuit breakers which are introduce according to the rule 5 in section 5.3.2. The substation configurations existing in this model are One and a Half Breaker, and Double Bus Double Breaker. The Breaker Table and Configuration Table regarding this test system are shown below:

Breaker Number	Near Station	Far Station	Breaker type	Breaker Status	Circuit #1	Circuit #2	Original Station
1	1	1	3	1	-1	-26	1
2	1	1	3	1	-1	-2	1
3	1	1	3	1	-2	-13	1
4	1	1	3	1	-26	-12	1
5	1	1	3	1	-12	-9	1
6	1	1	3	1	-9	-13	1
7	1	1	3	1	-26	-8	1
8	1	1	3	1	-8	-24	1
9	1	1	3	1	-24	-13	1
10	2	2	3	1	-4	-17	2
11	2	2	3	1	-10	-17	2
12	2	2	3	1	-18	-17	2
13	2	2	3	1	-10	-16	2
14	2	2	3	1	-16	-18	2
15	3	3	3	1	-3	-5	3
16	3	3	3	1	-6	-5	3
17	3	3	3	1	-1	-3	3
18	3	3	3	1	-1	-20	3
19	3	3	3	1	-6	-20	3
20	4	4	3	1	-9	-27	4
21	4	4	3	1	-27	7	4
22	4	4	3	1	-27	-22	4
23	4	4	3	1	5	-22	4
24	4	4	3	1	-8	-22	4
25	4	4	3	1	21	-22	4
26	5	5	3	1	-4	-15	5
27	5	5	3	1	-11	-15	5
28	5	5	3	1	-4	-14	5
29	5	5	3	1	-11	-14	5
30	6	6	3	1	-3	-19	6
31	6	6	3	1	-2	-19	6
32	6	6	3	1	-19	-10	6
33	1	1	3	1	-23	-6	1
34	1	1	3	1	-23	7	1
35	1	1	3	1	-24	-23	1
36	7	7	3	1	-11	-25	7
37	7	7	3	1	-12	-25	7

Tab. 6-1: Breaker Table Matrix

Notice:

There are only type 3 breakers being used in this conceptual test system. This is due to the fact this system is exactly the same as one used in reference [8].

Circuit Number	Near Station	Far Station	Near Station[1]	Near Meas. Status	Far Meas. Status	Load Flow Avai.	Inf.	Near Station Area	Far Station Area	Orig. Near Station	Orig. Far Station
1	1	3	1	1	1	1	1	1	1	1	3
2	1	6	0	1	0	1	0	1	1	1	6
3	3	6	0	1	0	1	0	1	1	3	6
4	2	5	2	1	1	1	1	1	1	2	5
5	3	4	3	1	1	1	1	1	1	3	4
6	3	1	0	1	0	1	0	1	1	3	1
7	4	1	0	1	0	1	0	1	1	4	1
8	1	4	1	1	1	1	1	1	1	1	4
9	1	4	1	1	1	1	1	1	1	1	4
10	6	2	0	1	0	1	0	1	1	6	2
11	5	7	0	1	0	1	0	1	1	5	7
12	1	7	0	0	0	0	0	1	1	1	7
13	1	0	0	0	0	0	0	0	0	1	0
14	5	0	0	0	0	0	0	0	0	5	0
15	5	0	0	0	0	0	0	0	0	5	0
16	2	0	0	0	0	0	0	0	0	2	0
17	2	0	0	0	0	0	0	0	0	2	0
18	2	0	0	0	0	0	0	0	0	2	0
19	6	0	0	0	0	0	0	0	0	6	0
20	3	0	0	0	0	0	0	0	0	3	0
21	4	0	0	0	0	0	0	0	0	4	0
22	4	0	0	0	0	0	0	0	0	4	0
23	1	0	0	0	0	0	0	0	0	1	0
24	1	0	0	0	0	0	0	0	0	1	0
25	7	0	0	0	0	0	0	0	0	7	0
26	1	0	0	0	0	0	0	0	0	1	0
27	4	0	0	0	0	0	0	0	0	4	0

Tab. 6-2: Configuration Table

6.2.2. IEEE Reliability Test System 1996 [14]

This Test system was previously introduced in section 4.2. Again, an overview of the system is shown in Fig. 6-2. The system consists of 24 substations, 90 circuits (66 lines and 24 buses) and 119 breakers.

[1] Refer to section 5.3.3.

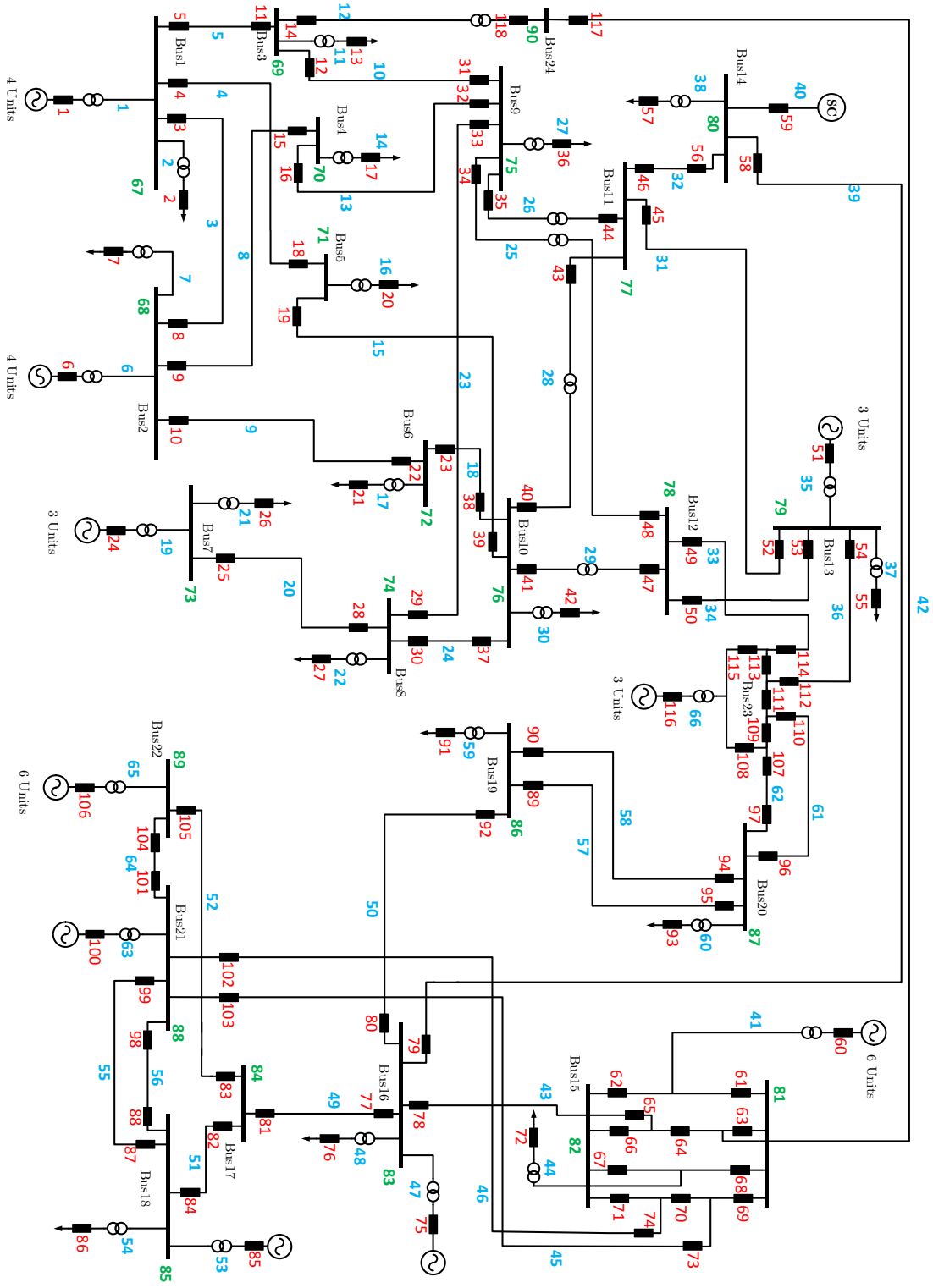


Fig. 6-2: IEEE Reliability Test System 1996

The Breaker Table Matrix and Configuration Matrix of this test system are large so they are provided in Appendix 1. This test system has all 4 types of breakers, and the most common configurations for a substation, i.e. One and a Half Breaker, Double Bus Double Breaker, and Ring.

6.3. Tested Scenarios, Results

For each test system, three consecutive switching scenarios are tested. Then the outputs of the topology engine are explained. In addition, the calculation time is also expressed and discussed in this section. The computer which the topology processer is running at has a 3.40GHz core. In addition, All of the outputs of the topology processor, in matrix form, are shown in the appendix as evidence of its execution.

6.3.1. Conceptual Model

The first switching pattern is shown if Fig. 3-2:

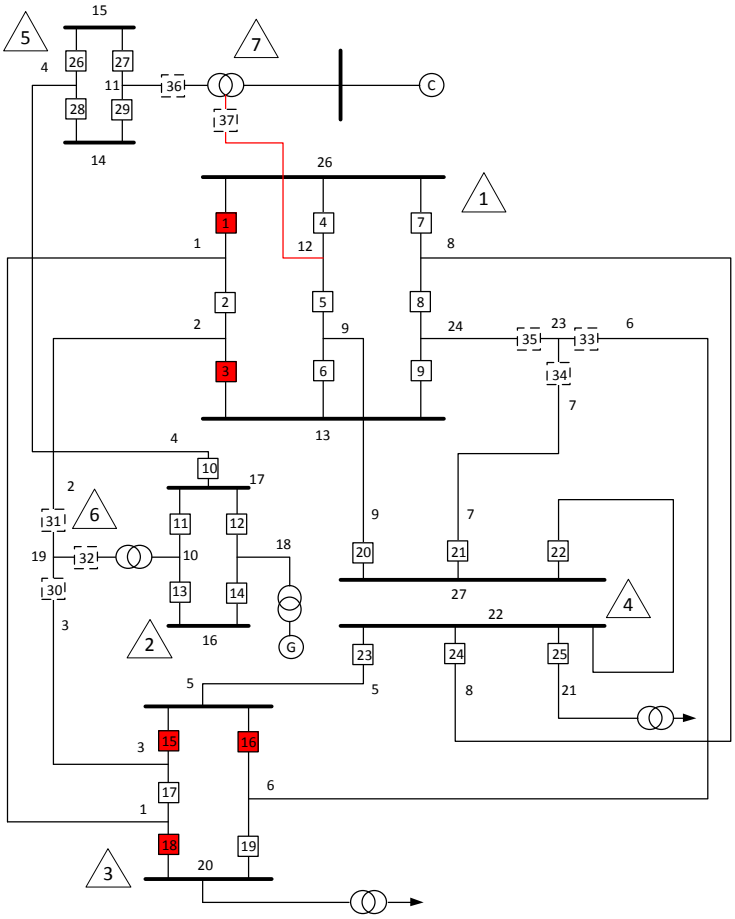


Fig. 6-3: Conceptual Test System, Scenario 1

What is the aim of this scenario?!

This scenario has a high order of complexity. 5 breakers are switched at the same time. One line has lost the measurements. This scenario will reveal the efficiency of the TP to deal with splitting stations, because both stations 1 and 3 are going to be split. In addition,

the ability of TP to deal with indirect disconnection of lines is going to be put into test. The indirect here means that the line is not disconnected because of its corresponding type 2 breaker is opened; it is disconnected because some open breakers within a substation has terminated the connection of the line and the station. So the line has one open end, which means it is indirectly disconnected (line 5 in this case). Also the ability of TP to deal with islanding is also going to be tested by this scenario as there will be two separate islands. So, this is a very complex scenario.

Computatiom Time: 4.6 mS

As it can be seen in Fig. 6-3, breakers 1, 3, 15, 16 and 18 are switched to the open position. Please note that line 12 is not open; however, the measurements are lost; by lost here we mean the measurement device is installed but no available. This means that the line should be treated as an open one when it comes to the islanding check. The bus branch model for Fig. 6-3 is shown in Fig. 6-4:

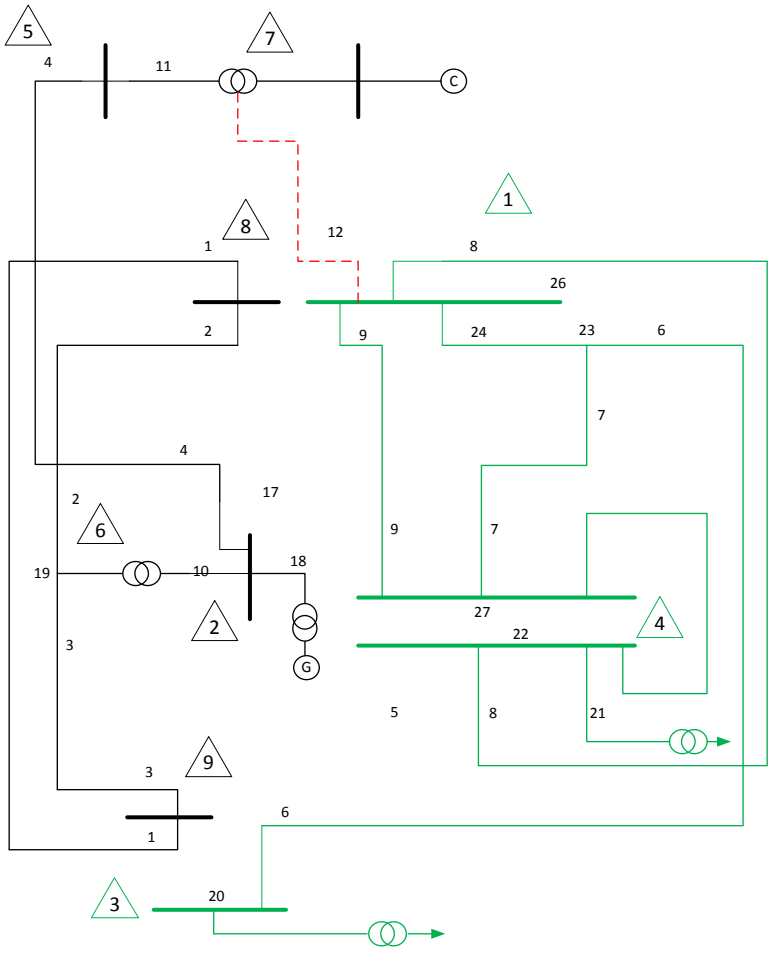


Fig. 6-4: Scenario 1, bus branch model

Notice:

When there are no measurements available for line 12, it is not open in reality. However, it should be treated like there is no link between substations 1 and 7 when it comes to check for separate islands.

The topology Processor is run and the results are as follows:

- Substations 1 and 3 are reported as stations with changes inside.
- For substation 1 two closed paths are found; one contains two: circuits 1 and 2 while the other contains remaining 6 circuits.
- For substation 3, three closed paths are found; one of the paths just contains a single circuit, number 5, which means it is disconnected. The other two contain two circuits each; 1 and 3, 6 and 20.
- Line 5 is reported as being disconnected; Also line 12 is reported as not be available for load flow.
- Substations 8 (lines 1, 2) and 9 (lines 1, 3) are automatically assigned numbers.
- Two separate islands are reported (black, and green) and assigned numbers automatically; Island 1 consists of stations 1, 3 and 4. Island 2 consists of stations 2, 5, 6, 7, 8 and 9.

Next switching scenario is performed as follow:

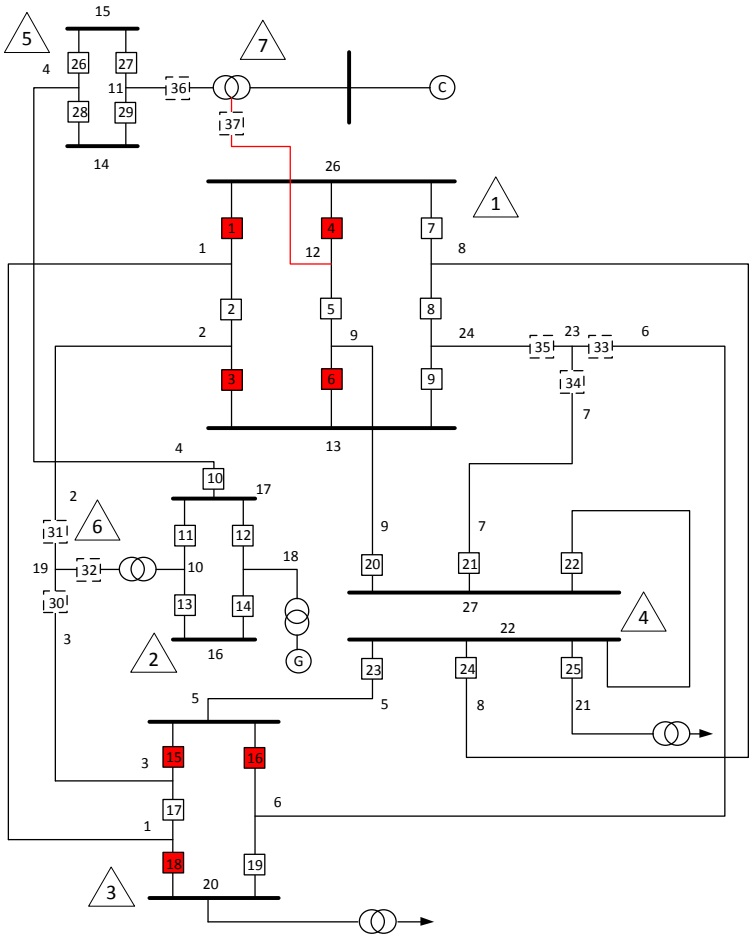


Fig. 6-5: Conceptual Test System, Scenario 2

What is the aim of this scenario?!

This scenario is going to show the efficiency of TP to deal with number assignation. Station 1 which has been split into two stations in previous scenario will be split into 3 stations in this scenario. This scenario will show that the number assigned the station derived from station 3 in previous cycle, won't change. Also, the scenario will show that TP has just performed the analysis for station 1, as that is the only station with changes compare to previous cycle (station 3 has open breakers, but won't be re-analyzed as the its breaker statuses remain the same).

Computatiom Time: 1.5 mS

It can be seen that the only change that has been made is to open breakers 4, and 6, i.e. there are only changes in substation 1. The bus branch mode is shown in Fig. 6-6:

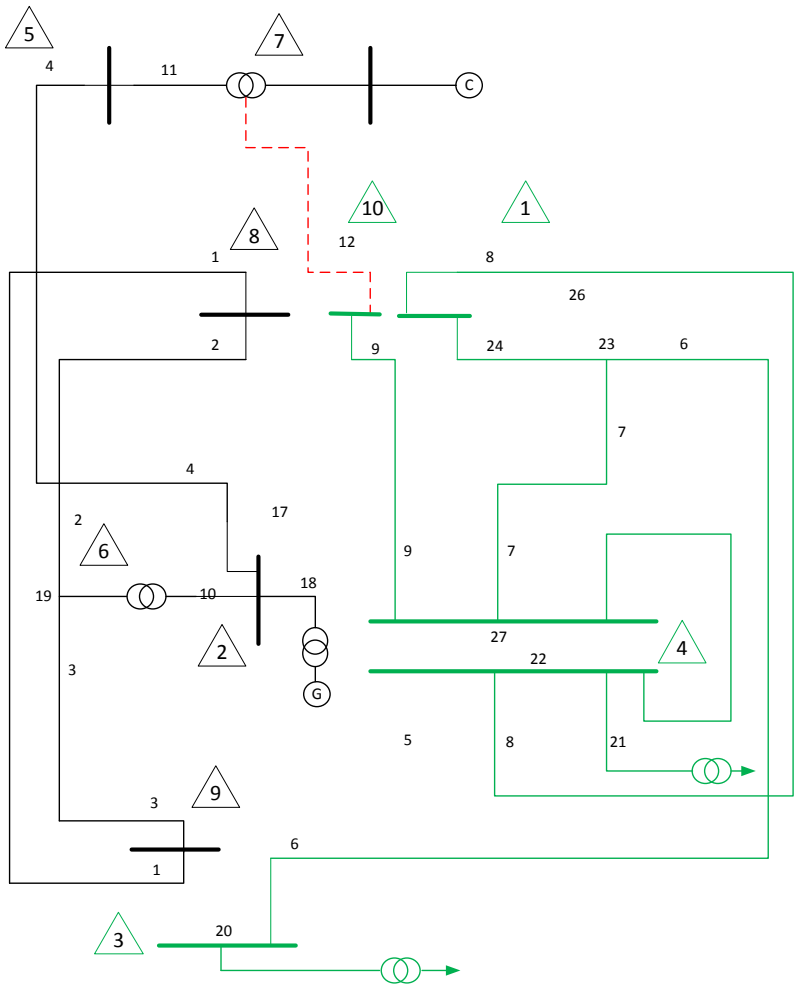


Fig. 6-6: Scenario 2, bus branch model

The following results are achieved:

- Substation 1 is reported as the only station with changes inside.
- Three closed paths are found. One has 4 different circuits: 8, 13, 24 and 26, while the other 2 have 2 circuits each: 1 and 2, 9 and 12.
- Line 12 is reported as not available for load flow.
- Substation 10 (circuits 9 and 12) are automatically assigned a number.
- Two separate islands are reported and assigned numbers automatically. Island 1 consists of stations 1, 10, 3 and 4. Island 2 consists of stations 2, 5, 6, 7, 8 and 9.

The final switching pattern is as shown in Fig. 6-7:

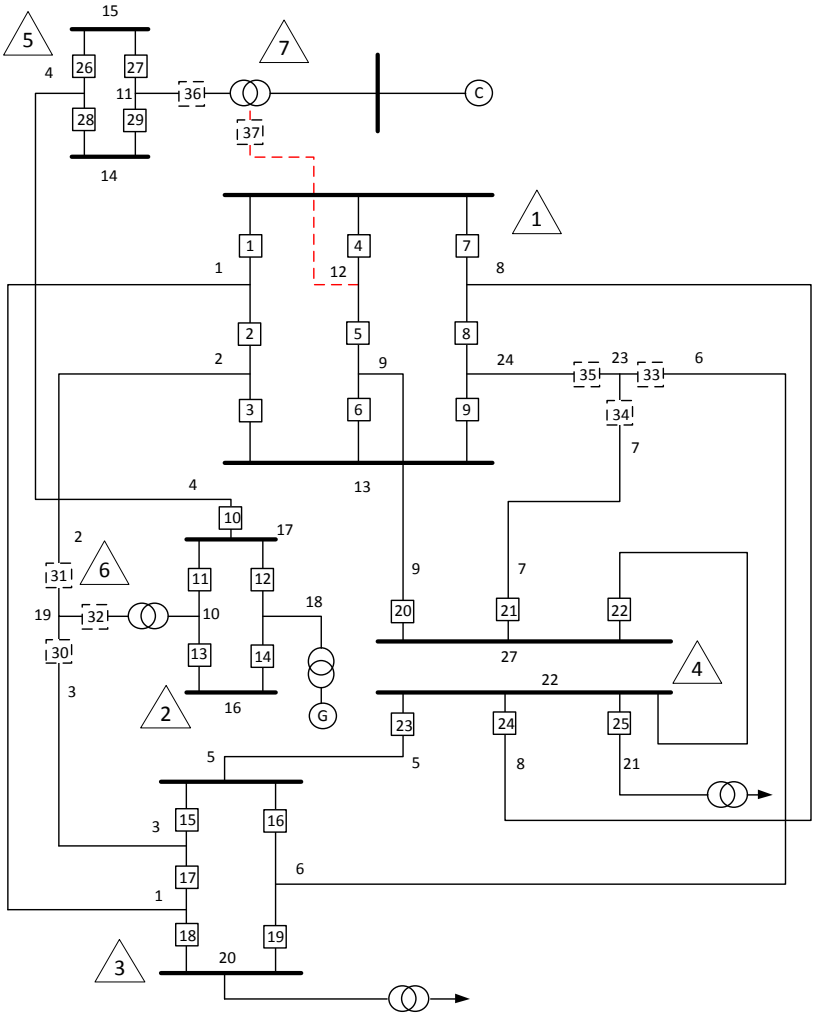


Fig. 6-7: Conceptual Test System, Scenario 3

What is the aim of this scenario?!

This scenario is going to show the efficiency of TP to deal with merging stations. All the breakers are closed so some stations in the system will merge. Also the number assignation will be tested when there are merging stations. Moreover, the ability of TP to detect the re-connected lines will be tested. There are 7 breakers in two stations whose statuses are changed at the same time.

Computatiom Time: 974 µS

All the open breakers are closed back. Here are the results:

- Stations 1 and 3 are reported with changes inside.
- In both stations just one single closed path can be found containing all the stations circuits.
- Stations 1, 8 and 10 are merged and automatically assigned number 1. Also stations 3 and 9 are also merged and assigned number 3.
- Line 5 is reported as connected again.
- There is just one island in the system containing all the system stations.

Also, the bus branch model is shown in Fig. 6-8 in the next page.

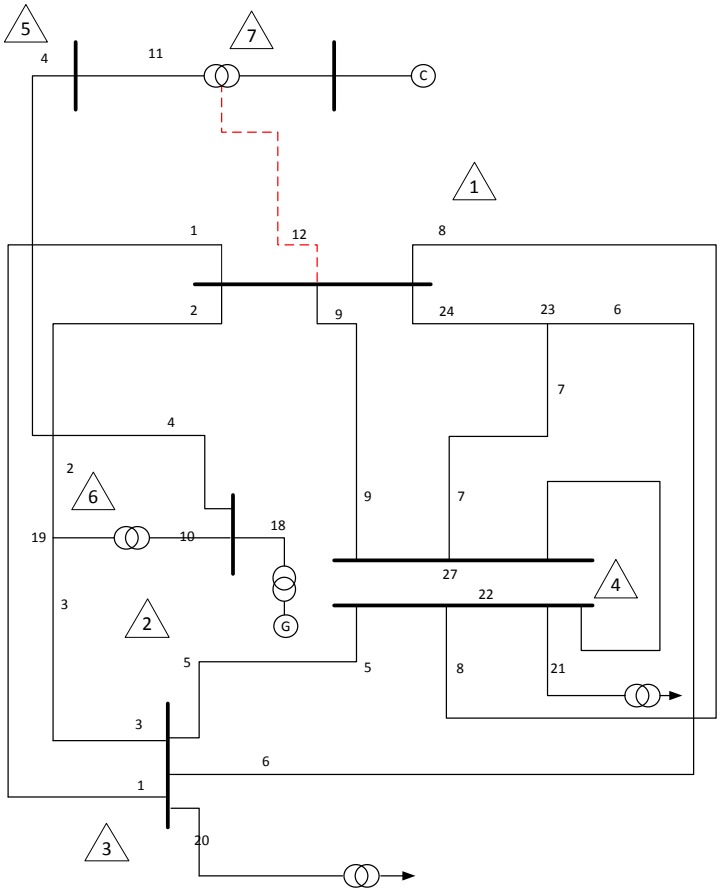


Fig. 6-8: Scenario 3, bus branch model

6.3.2. IEEE Reliability Tests System 1996

In Chapter 4, it is shown how the IEEE Reliability Test System 1996 was simulated in real-time using the Opal-RT real-time simulator. This has provided the possibility of having measurements very similar to a PMU data. Therefore, in this section, first the methods introduced in 5.3.5 are tested using a simple example, and then the same approach in section 6.3.1 is followed to show the efficiency of proposed topology engine.

Consider the following figure:

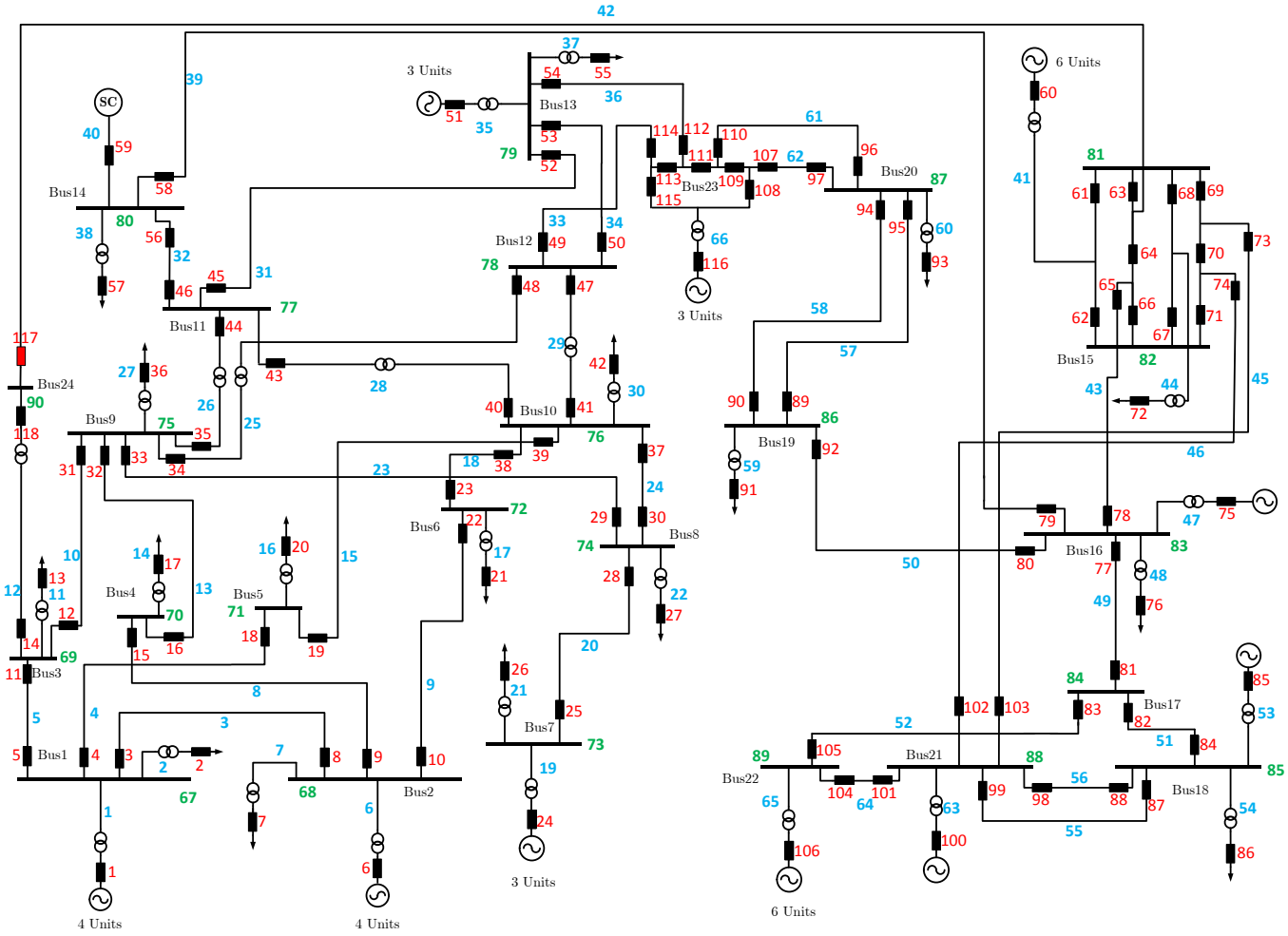


Fig. 6-9: Breaker 117 is switched

As it can be seen in the figure, breaker 117 is switched. The received measurements from the simulation are 50 samples per second; the Fig. 6-10 shows the measurements related to breaker 117, and its corresponding measurement 42:

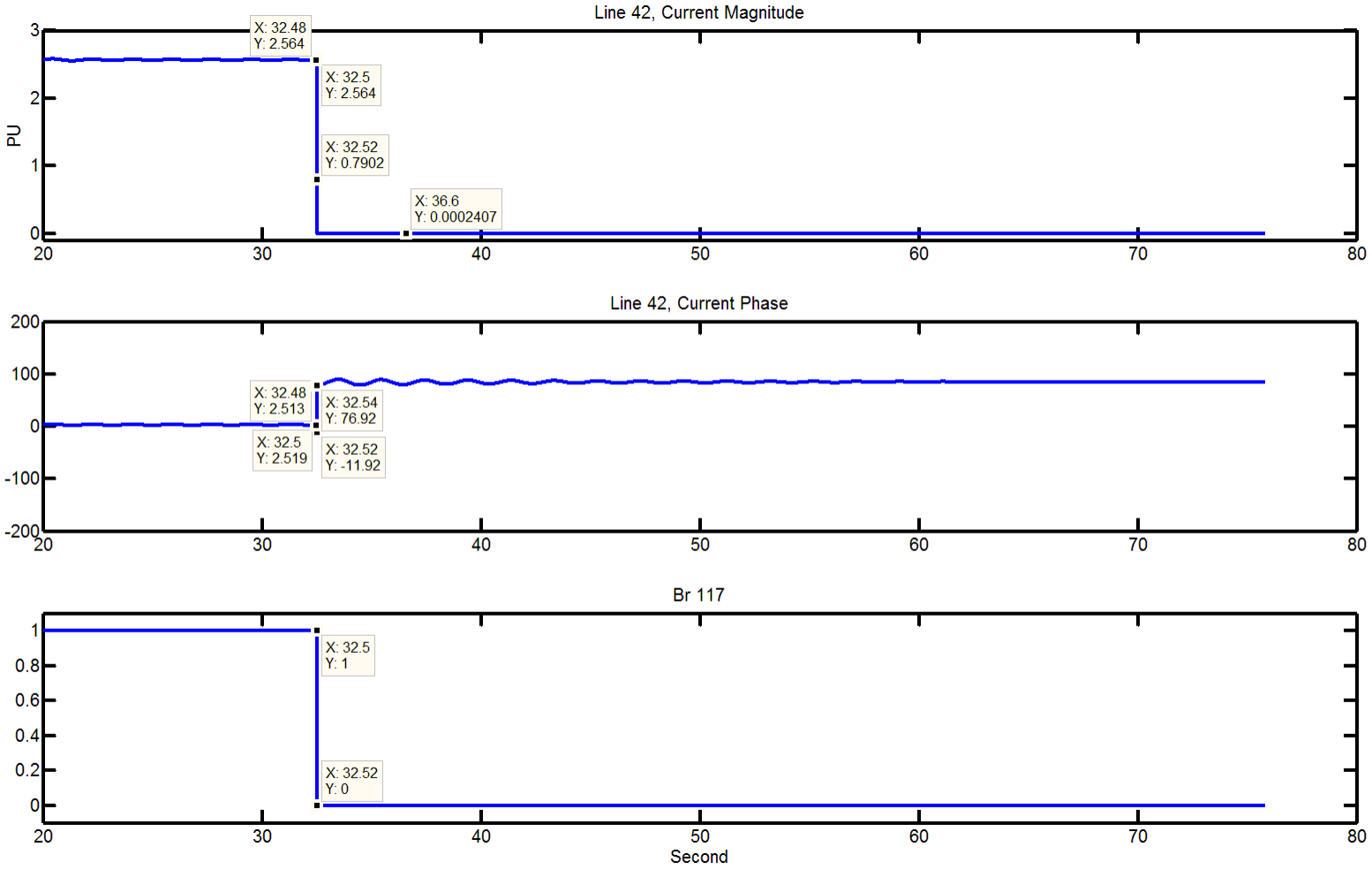


Fig. 6-10: Breaker 117 is switched, results

As it can be seen in the figure, the first snapshot of time that shows the breaker status change is *t=32.52*. Just one time stamp before, i.e. 32.5 shows that the breaker is closed. As the breaker opens, sudden changes can be seen in its corresponding line current both in magnitude and phase. The current magnitude suddenly decrease to almost 0; in the case of current phase, it shows a sudden decrease at the same time that the breaker switches, and then is has an increase step in the next snapshot, i.e. *t=32.54*. This initial decrease in current phase happens because of a transient undershoot. This is a kind of typical behavior among non-minimum phase dynamic systems, i.e. systems which have a zero on right half plane of zero-poles.

In the case that the change has happened within the substation, one of the busses current magnitude and phase should be investigated. As said previously, in this case the current magnitude and phase can just be used as verification method, and they can't find the breaker with changes independently.

- Magnitude Verification:

$$I^{32.52} - I^{32.5} = 0.7902 - 2.564 = -1.7738\ PU$$
$$I^{32.5} - I^{32.48} = 2.564 - 2.564 = 0$$

The current magnitude difference in time stamp 32.52 with its previous time stamp 32.5 is greater than 100 times of difference between magnitude at instances 32.5 and 32.48, which means that switching has happened at time 32.52.
In addition, if the digital measurement for breaker 117 is lost temporarily, the current magnitude is equal to 0.0002 which shows that the line is disconnected.

- Phase Verification:
As said before, the phase has a decrease at the same instant which switching has happened, and then has a sudden increase afterwards. Phase verification is done for both the instant the same as switching and also for one time stamp later:

$$\Phi^{32.5} - \Phi^{32.48} = 2.519 - 2.513 = 0.006$$
$$\Phi^{32.52} - \Phi^{32.5} = -11.92 - 2.519 = -14.439$$
or
$$\Phi^{32.54} - \Phi^{32.5} = 76.92 - 2.519 = 74.401$$

As it can be seen from the above calculations, the equation (5.2) is verified by an acceptable margin, so it detects the switch status has been changed.

After the breaker with changes is verified, or identified by current magnitude through line 42, it just reports that line 42 is disconnected.
Fig. 6-11 shows the bus branch model of the network:

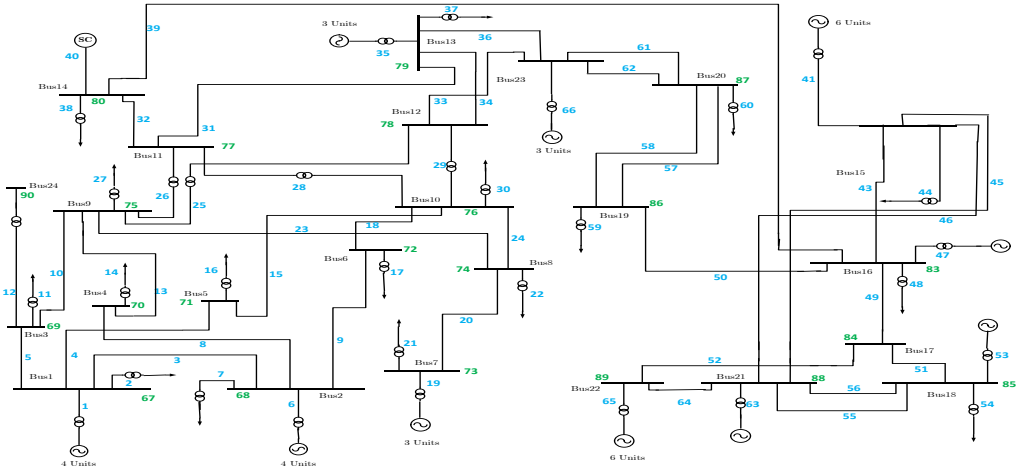


Fig. 6-11: Bus branch model

In the case that some switching happens within the substation, one of the buses which belongs to that substation can be used for magnitude phase verification. Consider the following figure:

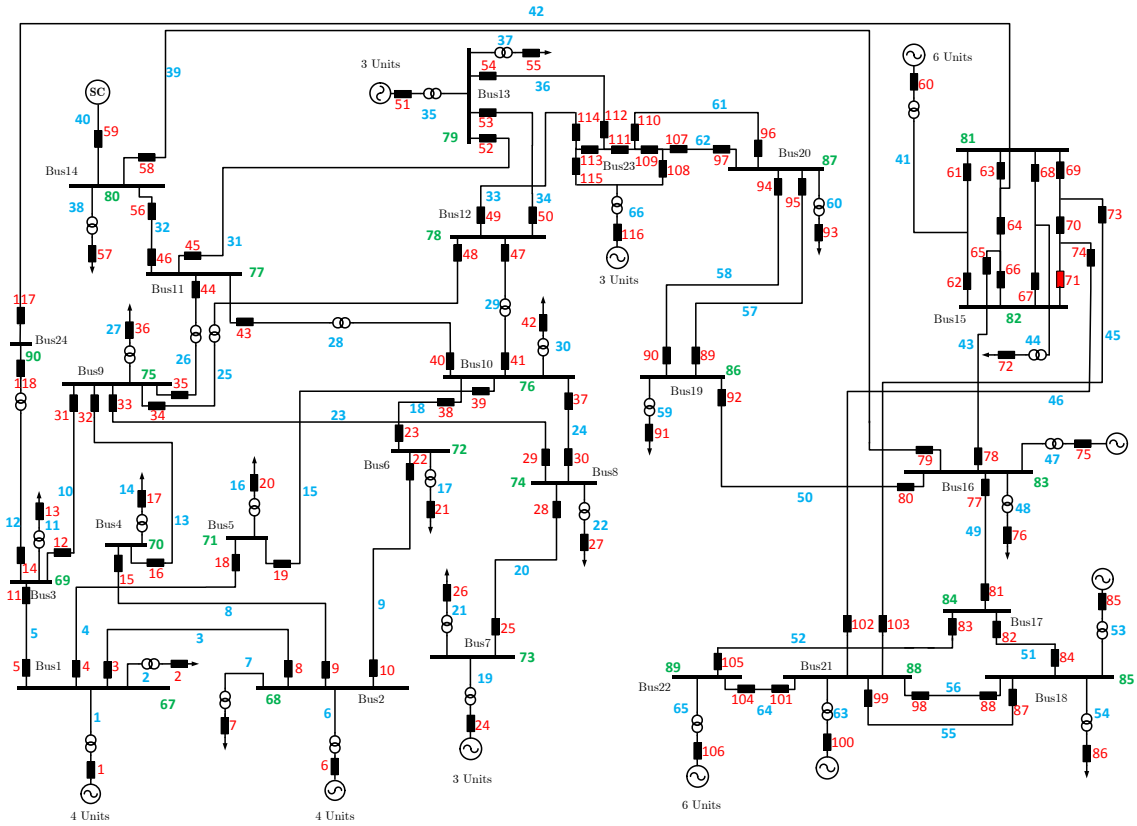


Fig. 6-12: Breaker 71 is switched

It can be seen in Fig. 6-12 that breaker 71 is switched. In this case, either bus 81, or bus 82 can be used for verification method. Here bus 81 is chosen, and the results are shown in Fig. 6-13 in the next page:

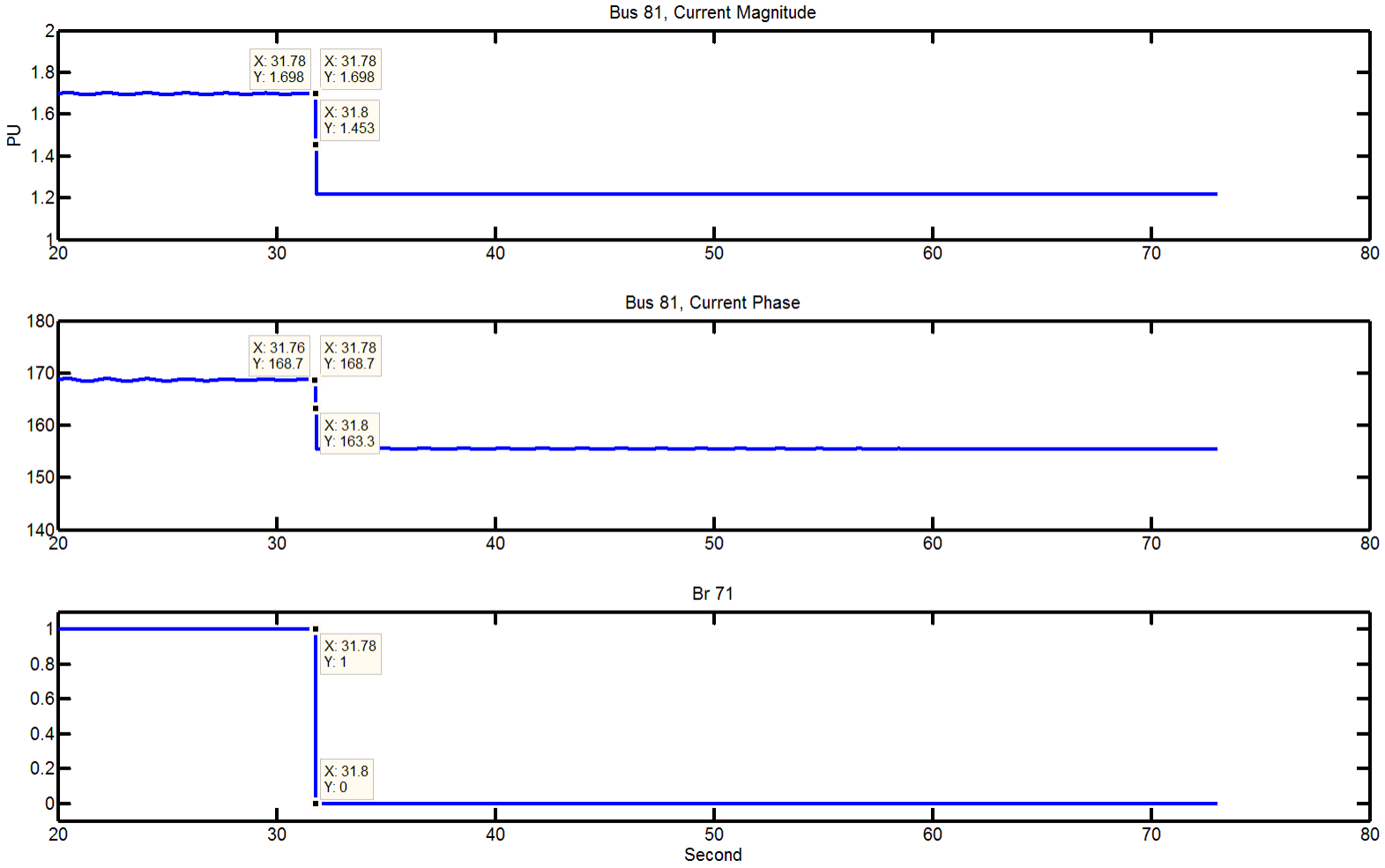


Fig. 6-13: Breaker 71 is switched, results

It is obvious from the figure that the breaker status changes at time t=31.8. Both the current magnitude and phase have a sudden decrease, and can be used for change verification:

- Magnitude Verification:

$$I^{31.8} - I^{31.78} = 1.453 - 1.6998 = -0.2468\ PU$$
$$I^{31.78} - I^{31.76} = 1.6998 - 1.6998 = 0$$

The current magnitude difference in time stamp 31.8 with its previous time stamp 31.78 is greater than 100 times of difference between magnitude at instances 31.78 and 31.76, which means that switching has happened at time 31.8.

- Phase Verification

$$\Phi^{31.8} - \Phi^{31.78} = 163.3 - 168.7 = -5.4$$

$$\Phi^{31.78} - \Phi^{31.76} = 168.7 - 168.7 = 0$$

So the switching occurrence within the substation is verified. As the switching has no effect in topology in this case, the bus branch model is as follow:

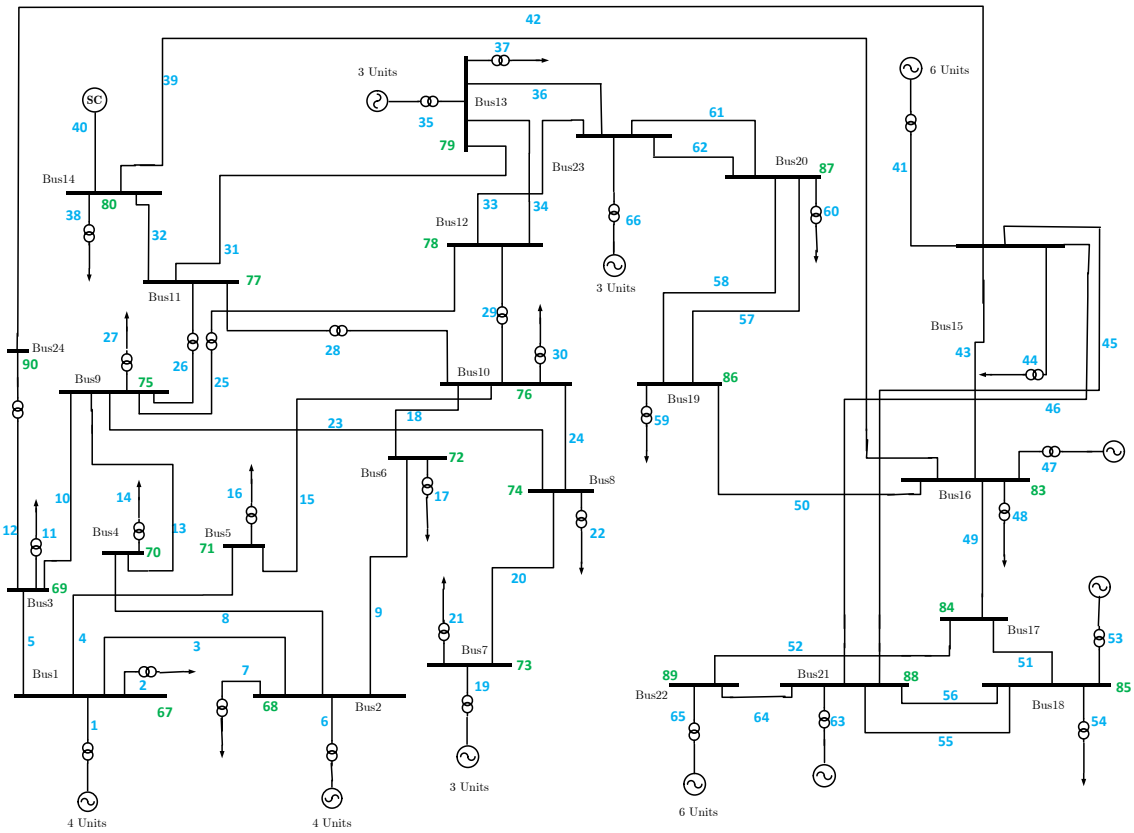


Fig. 6-14: Bus branch model

Having shown the usage of PMU measurement, next step is test the topology engine itself with this test system. Actually, as this system has three different substation configurations, and a sufficient amount of breakers, lines, and well detailed complexities, it is possible demonstrate the efficiency of the proposed topology engine.

First switching pattern is shown in Fig. 6-15:

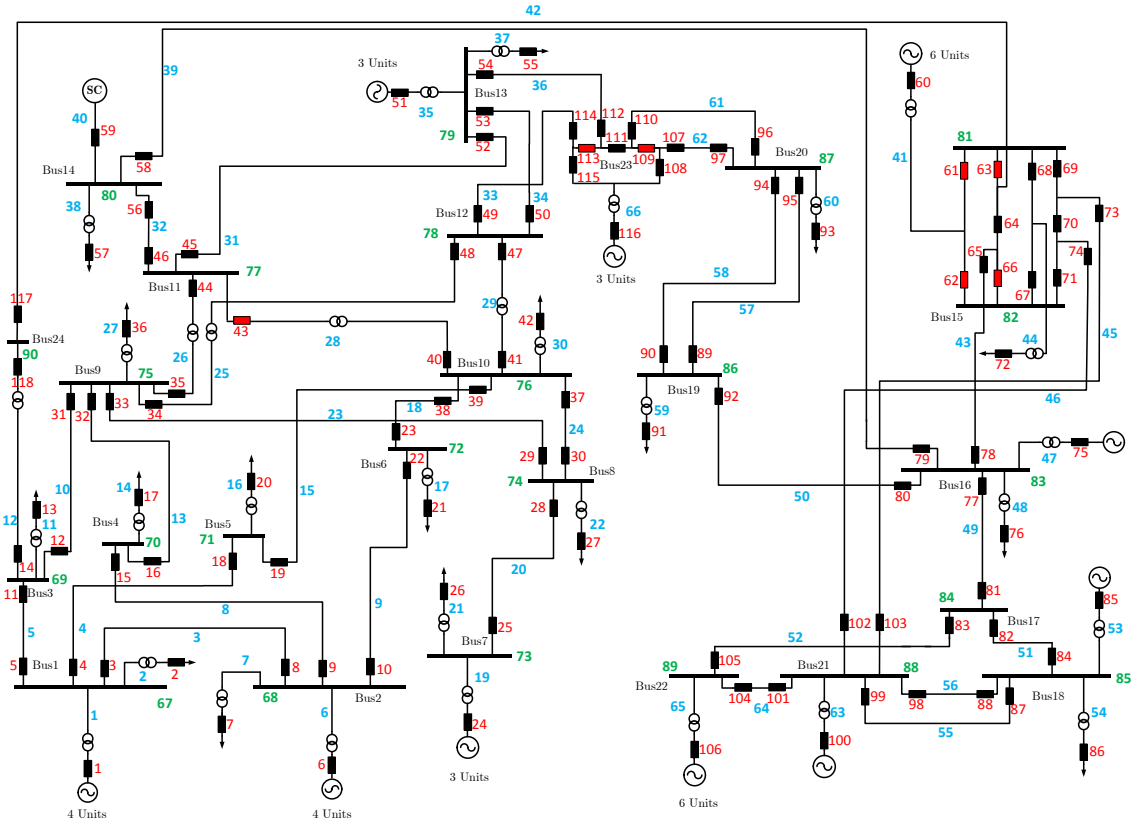


Fig. 6-15: 1996, first scenario

What is the aim of this scenario?!

This scenario has a high order of complexity. 7 breakers are switched at the same time including both type 1 and type 3 breakers. Two different stations are involved. This scenario will reveal the efficiency of the TP to deal with splitting stations, because both stations 15 and 23 are going to be split. In addition, the ability of TP to deal with both indirect and direct disconnection of lines is going to be put into test. The definition of direct disconnection and indirect disconnection is explained before.

Computation Time: 5.4 mS

From the figure it can be seen that breakers 43, 61, 62, 63, 66, 109, and 113 are opened. The bus branch model is shown in Fig. 6-16:

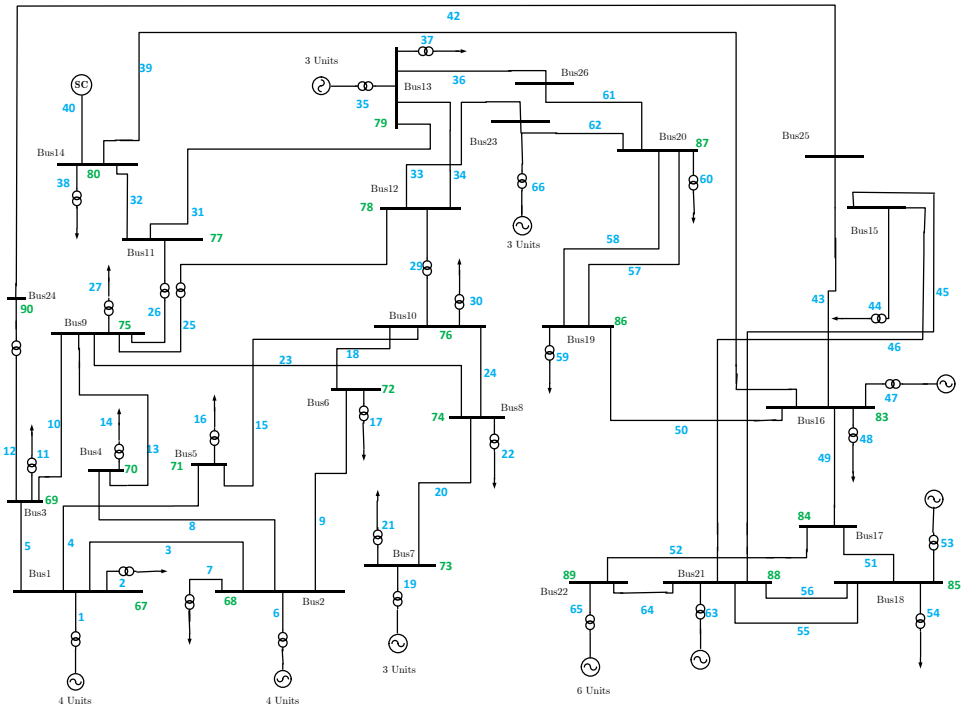


Fig. 6-16: 1996, first scenario Bus branch model

The following results are achieved after running the topology engine:

- Stations 15 and 23 are reported with changes inside.
- For station 15, 3 closed paths are found. One of them has just one single circuit 41 which means it is disconnected. One of them has two circuits 42 and 43 (station 25) while the other one has the remaining 5 circuits of the station.
- For station 23, 2 closed paths are found. One of them has two circuits 36 and 61 (station 26), while the other the one contain 3 circuits 62, 33, and 64.
- Line 28 is reported being connected again.
- There is just one island in the system containing whole the system stations.

Next switching pattern is shown in Fig. 6-17:

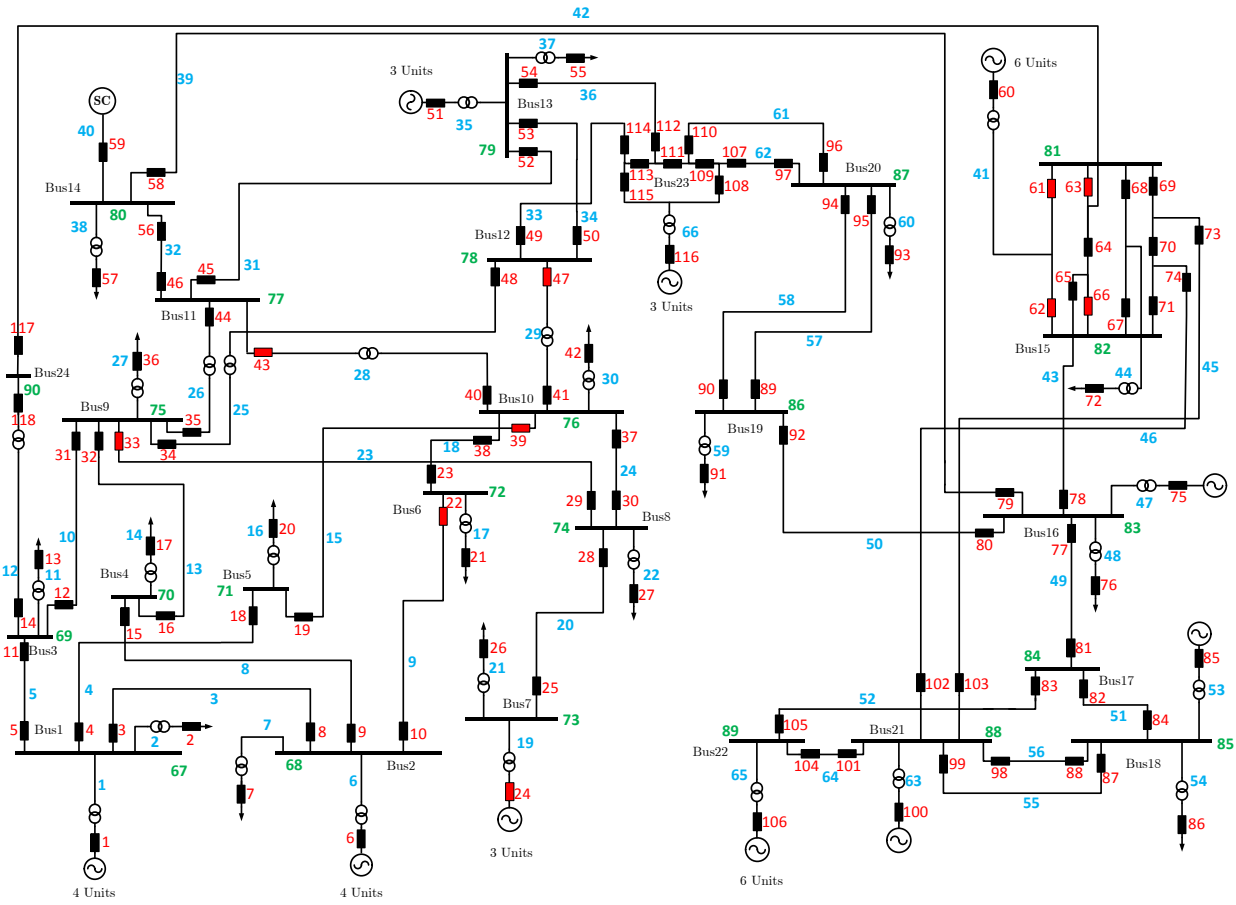


Fig. 6-17: 1996, second scenario

What is the aim of this scenario?!

This scenario is going to show the efficiency of TP to deal with merging stations and its corresponding number assignation. Station 23 which has been split into two stations in previous scenario will be merged again to one single station. Also, the scenario will show that TP has just performed the splitting/merging analysis for station 23, as that is the only station with changes compare to previous cycle (station 15 has open breakers, but won't be re-analyzed as the its breaker statuses remain the same). The islanding analysis ability of TP will also be analyzed as there will be two separate islands. Also, the energization check ability will be put into test, as one of the islands will be de-energized. The scenario has a high complexity as 7 breakers are switched at the same time.

Computation Time: 7.7 mS

As it can be seen from Fig. 6-17, breakers 33, 22, 24, 39, 47 are opened, while breakers 109, and 113 are closed back. The following figure shows the bus branch model:

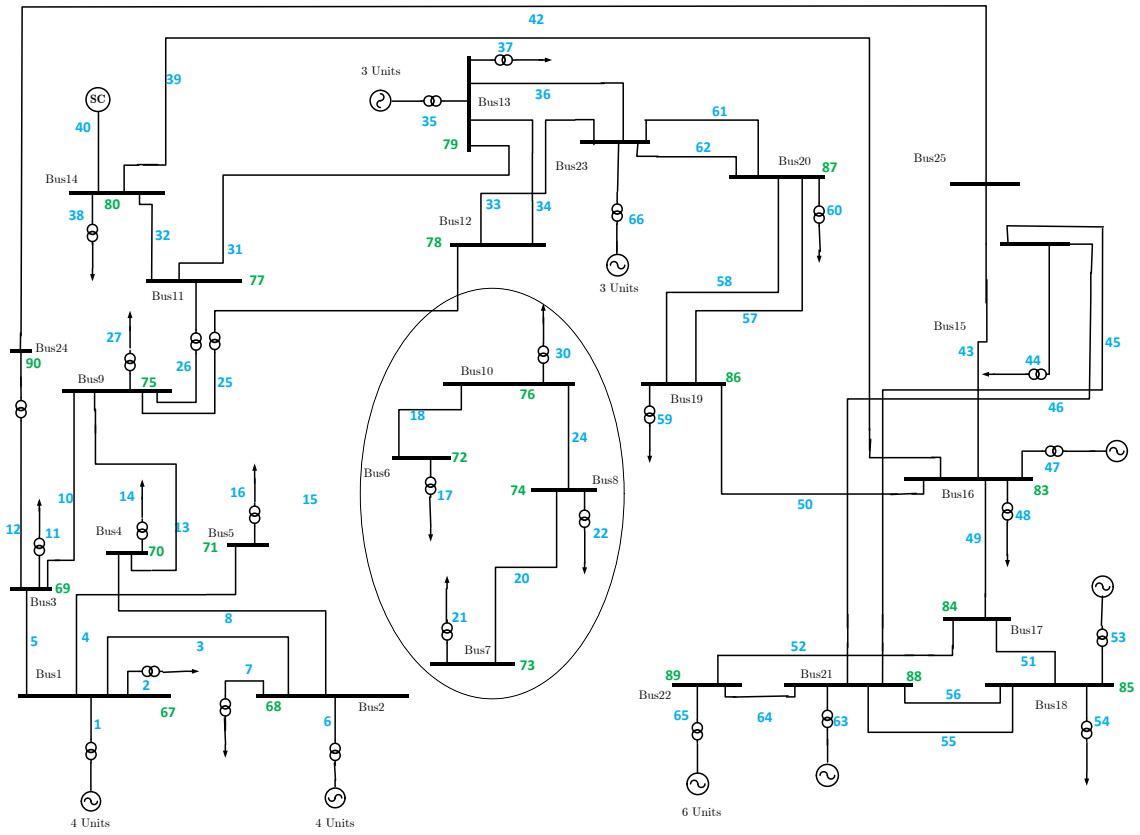


Fig. 6-18: 1996, second scenario Bus branch model

The output of the topology engine shows that:

- Station 23 is reported as the only one with changes inside.
- There is one single closed path containing all of station 23 circuits. That means substation 23 and 26 are merged back to original station (23).
- For station 23, 2 closed paths are found. One of them has two circuits 36 and 61, while the other the one contains 3 circuits 62, 33, and 64.
- Lines 23, 9, 29, 15, and 19 are reported as disconnected. Lines 28 and 41 have been disconnected previously.
- There are two islands in the system; one contains substation 6, 7, 8 and 10 and is de-energized. The other contains the remaining stations and is energized.

The final switching scenario is to close all the open breakers as shown in Fig. 6-19:

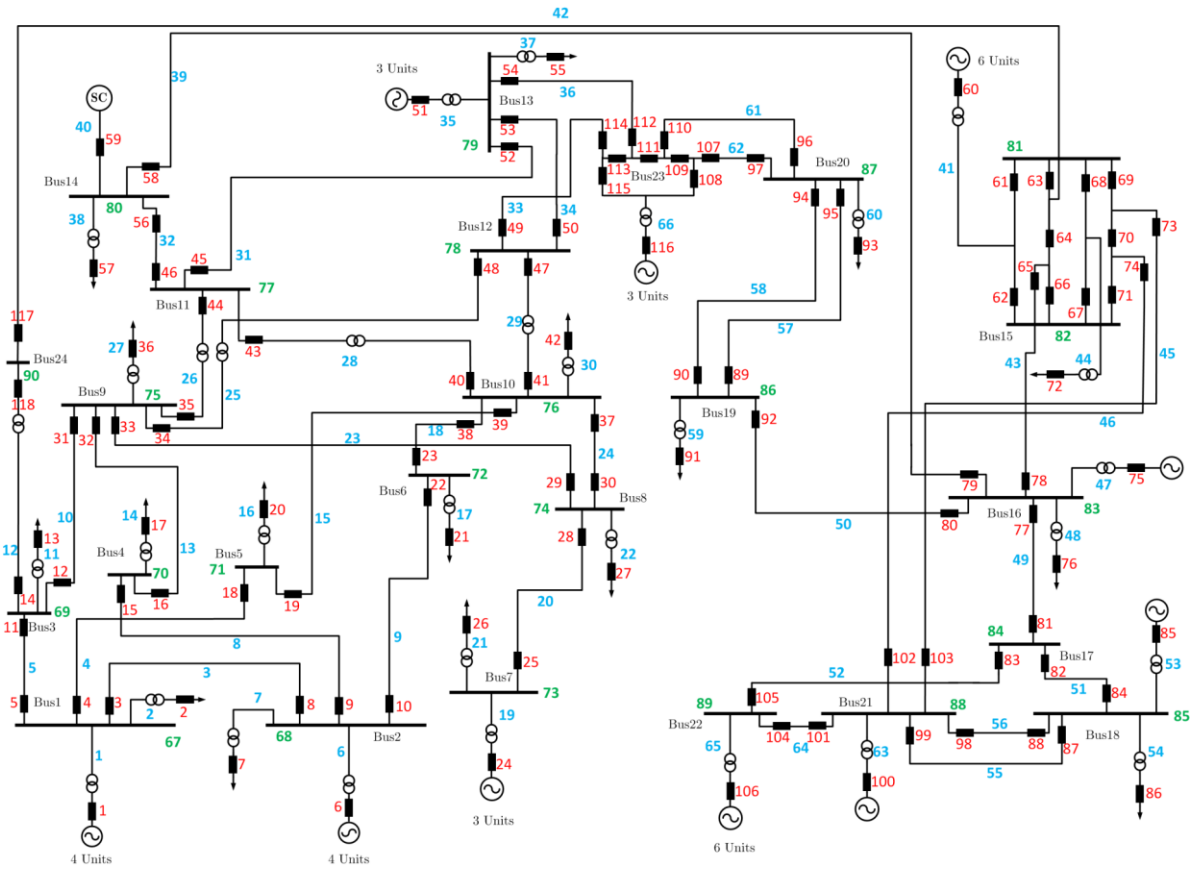


Fig. 6-19: 1996, third scenario

What is the aim of this scenario?!

This scenario has the highest complexity compare to all 5 previous ones. 10 breakers are switched at the same time. Station 15 will merge. Both directly and indirectly disconnected lines will be connected again. Two separate islands will be merged into one single energized one. The ability of TP to deal with all these changes is going to be put into test.

Computation Time: 3.7 mS

As all the open breakers are closed back, the bus branch model is as follow:

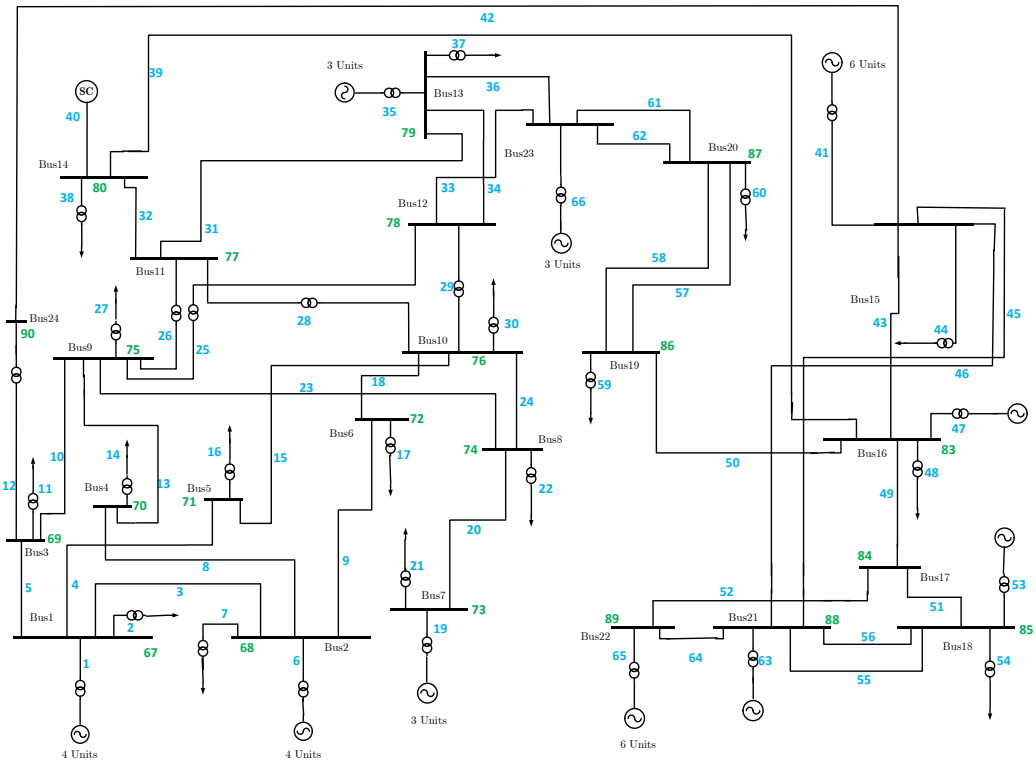


Fig. 6-20: 1996, third scenario bus branch model

The output of the topology processor is as follow:

- Station 15 is reported as having changes inside.
- For this station, just one single closed path is found which includes all its circuits.
- Stations 25 and 15 are merged and formed the original station 15.
- Lines 9, 19, 23, 15, 28, 29, and 41 are as reported being connected again.
- There is just one island in the system containing whole the system stations which is energized.

6.4. Computation Time Discussion

To the knowledge of the author, there are just two previous works which have presented the computation time in their results [10][16]. In [10], the author has just performed one test without providing any information about neither the test scenario nor the test system topology. He has mentioned 0.29 seconds as the computation time. In [16], there are different results regarding the computation time in the range of 20 to 150 micro seconds. However, the algorithm of [16] is tested in ABB industrial core, and no information is provided regarding the computation strength. Probably, the algorithm in [16] is tested with a far stronger CPU compare to the one of a typical PC. Another point here is that

the algorithm in [16] is graph based, very complex, and impossible to be implemented independently (e.g. for research purposes).
For the method in this thesis, the computation time is as below:

Scenario	Description	Computation Time
Conceptual Model, 1	5 breakers are opened simultaneously. 2 Stations suffer changes inside. Stations with changes are split to two stations each. 1 line is disconnected. There are two separate islands.	4.6 ms
Conceptual Model, 2	2 breakers are opened simultaneously. 1 Stations suffer changes inside. The station is split into 3 stations. There are two separate islands.	1.5 ms
Conceptual Model, 3	7 breakers are opened simultaneously. 2 Stations suffer changes inside. Station merging occurs. There is one island.	974μs
IEEE Model, 1	7 breakers are opened simultaneously. 2 Stations suffer changes inside. Stations with changes are split to two stations each. 2 lines are disconnected. There is one island.	5.4 ms
IEEE Model, 2	7 breakers are opened simultaneously. 1 Stations suffer changes inside. Station merging occurs. 7 lines are disconnected. There are two separate islands.	7.7 ms
IEEE Model, 3	10 breakers are opened simultaneously. 1 Station suffers changes inside. Station merging occurs. 7 lines are reconnected. There is one island.	3.7 ms

As it can be observed from the above table, the order of computation time is in order if milliseconds. However, the complexity of scenarios is higher than reality as several breakers are switching at the same time. Also, simultaneous changes happen at different parts of the system. In reality, the probability of suffering such a pattern of simultaneous changes is low. Therefore, the scenarios here can be considered as having high order of complexity. Actually, several additional scenarios with change of one single breaker were tested; the computation time for all was in the order of several hundred microseconds.

In addition, the CPU used for performing TP in this thesis is a 3.4GH CPU of a typical PC using MATLAB in windows. The computation time will further decrease using a commercial platform for performing TP calculations. In addition, running each part of the system on an independent core (multi-core calculation) will significantly decrease the computation time.

Considering all the mentioned facts, the computation time achieved in this thesis is acceptable (comparing with previous works).

6.5. Chapter Summary and Conclusion

In this chapter, test results are completely provided. First, the two different test systems are introduced, and then the result of test regarding to each one is demonstrated. In addition, the measurements from the real-time simulation of IEEE Reliability Test System 1996 are provided and illustrations on the use of PMU data are given.
The results demonstration and discussion proves the efficiency of the proposed TP to deal with different types of scenarios regarding power network topology changes. All the aspects of a typical topology engine are tested and scenarios are designed in a way to cover most of the critical and important situations regarding the area of topology processing.

Chapter 7: Conclusion and Future Work

7.1. Summary

In this thesis, the field of topology processing in power systems was fully scrutinized. Topology processing refers to a crucial function in real-time modeling of power systems, in which the topology of the system is updated automatically using the received data of breakers status, and a static data base of network connectivity parameters. To be able to perform any kind of network analysis algorithm, it is necessary to determine the system topology first as all the mathematical equations of the system are constructed on its current topology. Specifically, topology processing plays a very important role in any state estimator. As state estimator is the key function in any Energy Management System (EMS), it is of high importance to have a topology processor which is robust enough and can deal with any kind of switching scenarios in power systems.

With the arrival of Phasor Measurement Units (PMUs), many of the power system researches' focus shifted from application of traditional measurements to application of PMUs in power systems. PMU directly measures the states of the system, i.e. voltage magnitude and angle; it also measures the current phasors. This is a very high advantage as previously it was just only possible to measure the voltage and current magnitude, or complex powers. The ability of measuring the states directly will highly increase the system operator awareness of what is going on in the system. Also, PMU provides more accurate measurements with a much higher rate compare to conventional telemetered measurements, i.e. around 100 samples per second compare to one sample every seconds. The measurements received from PMU are time stamped with a very high accuracy, using the Global Positioning System (GPS); this provides the possibility of gathering data from different parts of a power system and then synchronizes them so as monitor all parts of a huge system at the same time. This is called wide area monitoring and is an efficient way of monitoring power systems.

Like many other applications, there are proposed algorithms for state estimation that use PMU data. As topology processing is an integral part in any state estimator, there is a need to have a robust topology processor that works with PMU data also. Please note that if a topology processor works efficiently with conventional data, with some slight modifications it can also work with PMU data; however, this is not a reversible meaning that if a topology processor can only handle PMU data, it may not work with traditional

measurements. Therefore, in this thesis an algorithm that used traditional data is proposed, and then the usage of PMU data is considered in such an algorithm.

The theory of state estimation was discussed both for traditional state estimators, and for PMU-only SEs. After a certain familiarity with state estimation is achieved and the necessity of a topology engine in state estimation become apparent, the theory of topology processing is explained in well details. Then a literature review is carried out, and the deficiencies and defects of all previous articles are revealed and discussed. Finally a proposed algorithm is fully developed, and is explained by a combination of descriptive explanations, flowchart and snippets of MATLAB code. The algorithm is tested over two different tests systems, and all possible switching patterns are tested. The tests results show that the algorithm works efficiently and all of the limitations of previous work are covered.

In addition, the IEEE Reliability Test System 1996 is simulated in real-time using Opal-RT real-time simulator; this provides the ability to emulate PMU data, and discuss the integration of PMUs in the proposed method for topology processing. Also, the existence of a simulated network in real-time with sufficient degree of complexity is a good step towards the development of a platform to implement a full PMU-only state estimator, using the proposed topology processor.

7.2. Conclusion

From the results which are shown in Chapter 6, it can be observed that the proposed algorithm of this thesis is quite efficient in dealing with several different topological situations. This includes the ability to perfectly detect and report splitting or merging stations, and the well-operating number assignation.

The proposed methods have all the necessary parts of a typical topology processor, and no part is neglected. Moreover, it performs the topology processor just for those parts of the system which have suffered changes compare to previous execution cycle. The outputs are in a form which makes the changes in the topology to be easily traced.

Also, the possibility of application of PMU data is covered, and relative issues are fully discussed. The illustrations of different situation by simulated real-time waveform make the explanations and procedure rigorous.

In addition, the algorithm has fully covered the mentioned drawbacks of the other works in the area, and has demonstrated a very good operability. The procedure is clear, and straightforward, and rigorous. The well-detailed explanation of the algorithm along the flowcharts and snippets of MATLAB codes make any independent implementation easy.

As a conclusion, the proposed method of this thesis is a quite robust TP, which is able to deal with any kind of topological changes in the system. It has the ability to work with TCP/IP data, or PMU-only data. Also, it can exploit the PMU data to verify those of

TCP/IP. The method is a complete practical one, and can be used further for PMU-only state estimators.

7.3. Contribution

Considering all the mentioned drawbacks with previous works in the area, and all the relative issued which are discussed, the proposed method in this thesis has the following contribution:

1- The only openly available TP that has the ability to work with different kinds of substations configurations. It can manage changes within the substation in an efficient way. This claim has been supported by provided test results, MATLAB snippets, and simulated waveforms.

2- The only TP which is considered to work with traditional data, or in PMU-assisted mode, or PMU-only. The discussion over this issue is rigorous, and supported by simulated waveforms and results.

3- The algorithm is coded in a way to work with any kind of power systems with slight modifications. The code is not just written for one test system. Several tests with two different systems are performed, and strong visual and numerical results are illustrated.

4- In a contrast with previous works in the area, the provided algorithm considers all the aspects of a practical topology processor. This includes number assignation, or energization check. It explains the procedures step by step, supporting the explanation by flowchart and MATLAB code snippets. This has made any independent implementation easier that previous works.

5- To the knowledge of the author, the proposed method of this thesis is the only openly available TP which determine the system topology by analyzing only those parts which have suffered changes. This makes tracking of the changes easier.

6- The thesis discusses the computation time. Considering the provided results, it has the potential to be used in a PMU-only state estimator. For a PMU-only SE all the calculations should be done in less than 20 ms.

7- The complete implementation code may be distributed to interested parties in conditions to be specified.

7.4. Future Work

As the real-time simulated model of IEEE Reliability Test System 1996 is available, there is a possibility to develop a state estimator which is only based on PMU measurements. This will involve a variety of future works such as:

- ✓ Developing a graphical output for the topology processor.
- ✓ Developing a real time automated model equation generator using the output of the topology processor.
- ✓ Investigating the effects of measurements transmission delays in the output.
- ✓ Develop and test the Phasor State Estimation method using the real-time simulated model and the proposed topology processor algorithm.
- ✓ Investigate the effects of breaker failure or wrong data on the proposed topology processor and suggests any possible solutions.

References

[1] F.C. Schweppe and E.J. Handschin, "*Static State Estimation in Electric Power Systems*," IEEE Trans., Vol. 62, NO. 7, July 1974.

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

[3] L. Vanfretti and J. H. Chow, "*Synchrophasor Data Application for Wide-Area Systems*", IEED PSCC, September 2010.

[4] A.G. Phadke, J.S. Thorp, and K. Karimi, "*State Estimation with Phasor Measurement*", IEEE transactions on Power Systems, 1986.

[5] L. Vanfretti, J.H. Chow, S. Sarawgi, and B.B. Fardanesh, "*A Phasor-data-based State Estimator incorporating phase bias correction*", IEEE Transactions on Power Systems, 2010

[6] A. Monticelli, "*State estimation in electric power systems: a generalized approach*", Springer, 1999.

[7] A.G. Phadke and J.S. Thorp, "*Synchronized Phasor Measurements and their Applications*", Springer, New York, 2008.

[8] A.M. Sasson, S.T. Ehrmann, P. Lynch, and L.S. Van Slyck, "*Automatic Power System Network Topology Determination*", IEEE Journals, Vol. PAS-92, 1973.

[9] A. Bose and K.A. Clements, "*Real-time Modeling of Power Networks*", IEEE Journals, Vol. 75, 1987.

[10] A. Bose and M. Prais, "*A Topology Processor That Tracks Network Modifications Over Time*", IEEE transactions on Power Systems, Vol. 3, 1988.

[11] Z. Yongli, T.S. Sidhu, M.Y. Yang, and L.M. Huo, "*An AI-based automatic power network topology processor*", www.paper.edu.cn.

[12] C. Qian, Z. Wang, and J. Zhang, "*A New Algorithm of Topology Analysis Based on PMU Information*", IEEE Conferences, 2010.

[13] O. Alsac, N. Vempati, B. Stott, and A. Monticelli, "*Generalized State Estimation*", IEEE Conferences, 1997.

[14] C. Grigg *et al*, "*The IEEE Reliability Test System-1996. A report prepared by the Reliability Test System Task Force of the Application of Probability Methods Subcommittee* ", IEEE Transactions on Power Systems, Vol. 14, 1999.

[15] Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommaendations Apr. 5, 2004, U.S.-Canada Power System Outage Task Force

[16] R. Kussel, R. Chrustowski, and C. Jaborowicz, "*The Topology Engine: A new Approach To Initializing And Updating The Topology Of An Electrical Network*", ABB Netzleittechnik GmbH, Germany.

[17] "*SmarTS Lab*" available at *http://www.vanfretti.com/Dr._Luigi_Vanfrettis_Website/Research_Group.html*

[18] K.A. Clements, "*Observability methods and optimal meter placement*", International Journal of Electrical and Energy Systems, 12(2):88-93, April 1990.

[19] B. Gou and A. Abur "*An Improved Measurement Placement Algorithm for Network Observability ability*", IEEE transactions on Power Systems, Aug 1998.

[20] T. Van Cutsem and P.J. Gailly, "*A Simple Algorithm for Power System Observability Analysis and Related Functions*", IFAC Symposium 39-83, Italy, 1983.

[21] A. Monticelli and A. Garcia, "Reliable Bad Data Processing for Real-Time State Estimation", IEEE Transactions on Power Apparatus and Systems, Vol. PAS-102, No. 3, pp. 1126 – 1139, 1983.

[22] K.A. Clements and P. W. Davis, "Multiple Bad Data Detectability and Identifiability: a Geomatric Approach", IEEE Transactions on Power Apparatus and Systems, Vol. PAS-98, pp. 1645 – 1652, Sept. 2979.

[23] H. M. Merril and F. C. Schweppe, "Bad Data Suppression in Power System Static Estimation", IEEE Transactions on Power Apparatus and Systems, Vol. PAS-90, pp. 2718-2725, Nov./Dec. 2971.

[24] ISO NEW ENGLAND PLANNING PROCEDURE NO. 9 APPENDIX B, "*Transmission Owners and ISO-NE Substation Bus Arrangement Guidline Working Group Report",* NEPOOL Reliability Committee, April 4, 2006.

[25] M.S. Almas, "*PMU-Assisted Local Optimization of The Coordination Between Protective Systems and Reactive Power Compensation Devices",* M.Sc. Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2011.

[26] "SimPowerSystems User's Guide" by "The MathWorks" available at *http://www.mathworks.com/help/releases/R13sp2/pdf_doc/physmod/powersys/powersys.pdf*

[27] Brochure for OPAL-RT simulator is available at *http://www.opal-rt.com/sites/default/files/brochure-eMEGAsim-120507.pdf*

[28] RT-Lab Professional available at *http://www.opal-rt.com/product/rt-lab-professional*

[29] J. Lu, D. Xie, Q. Ai, "*Research on Smart Grid in China*", IEEE T&D Asia, 2009.

[30] Cholesky Decomposition method available at *http://en.wikipedia.org/wiki/Cholesky_decomposition*

Appendix

A.1.

Breaker Number	Near Station	Far Station	Breaker type	Breaker Status	Circuit #1	Circuit #2	Original Station
1	1	1	1	1	0	0	0
2	1	1	4	1	0	0	0
3	1	2	2	1	3	0	0
4	1	5	2	1	4	0	0
5	1	3	2	1	5	0	0
6	2	2	1	1	0	0	0
7	2	2	4	1	0	0	0
8	2	1	2	1	3	0	0
9	2	4	2	1	8	0	0
10	2	6	2	1	9	0	0
11	3	1	2	1	5	0	0
12	3	9	2	1	10	0	0
13	3	3	4	1	0	0	0
14	3	24	2	1	12	0	0
15	4	2	2	1	8	0	0
16	4	9	2	1	13	0	0
17	4	4	4	1	0	0	0
18	5	1	2	1	4	0	0
19	5	10	2	1	15	0	0
20	5	5	4	1	0	0	0
21	6	6	4	1	0	0	0
22	6	2	2	1	9	0	0
23	6	10	2	1	18	0	0
24	7	7	1	1	0	0	0
25	7	8	2	1	20	0	0
26	7	7	4	1	0	0	0
27	8	8	4	1	0	0	0
28	8	7	2	1	20	0	0
29	8	9	2	1	23	0	0
30	8	10	2	1	24	0	0
31	9	3	2	1	10	0	0
32	9	4	2	1	13	0	0
33	9	8	2	1	23	0	0
34	9	12	2	1	25	0	0
35	9	11	2	1	26	0	0
36	9	9	4	1	0	0	0
37	10	8	2	1	24	0	0
38	10	6	2	1	18	0	0
39	10	5	2	1	15	0	0
40	10	11	2	1	28	0	0
41	10	12	2	1	29	0	0
42	10	10	4	1	0	0	0
43	11	10	2	1	28	0	0
44	11	9	2	1	26	0	0

45	11	13	2	1	31	0	0
46	11	14	2	1	32	0	0
47	12	10	2	1	29	0	0
48	12	9	2	1	25	0	0
49	12	23	2	1	33	0	0
50	12	13	2	1	34	0	0
51	13	13	1	1	0	0	0
52	13	11	2	1	31	0	0
53	13	12	2	1	34	0	0
54	13	23	2	1	36	0	0
55	13	13	4	1	0	0	0
56	14	11	2	1	32	0	0
57	14	14	4	1	0	0	0
58	14	16	2	1	39	0	0
59	14	14	1	1	0	0	0
60	15	15	1	1	0	0	0
61	15	15	3	1	-41	-81	15
62	15	15	3	1	-41	-82	15
63	15	15	3	1	-42	-81	15
64	15	15	3	1	-42	-43	15
65	15	16	2	1	43	0	0
66	15	15	3	1	-43	-82	15
67	15	15	3	1	-44	-82	15
68	15	15	3	1	-44	-81	15
69	15	15	3	1	-45	-81	15
70	15	15	3	1	-45	-46	15
71	15	15	3	1	-46	-82	15
72	15	15	4	1	0	0	0
73	15	21	2	1	45	0	0
74	15	21	2	1	46	0	0
75	16	16	1	1	0	0	0
76	16	16	4	1	0	0	0
77	16	17	2	1	49	0	0
78	16	15	2	1	43	0	0
79	16	14	2	1	39	0	0
80	16	19	2	1	50	0	0
81	17	16	2	1	49	0	0
82	17	18	2	1	51	0	0
83	17	22	2	1	52	0	0
84	18	17	2	1	51	0	0
85	18	18	1	1	0	0	0
86	18	18	4	1	0	0	0
87	18	21	2	1	55	0	0
88	18	21	2	1	56	0	0
89	19	20	2	1	57	0	0
90	19	20	2	1	58	0	0
91	19	19	4	1	0	0	0
92	19	16	2	1	50	0	0
93	20	20	4	1	0	0	0
94	20	19	2	1	58	0	0
95	20	19	2	1	57	0	0
96	20	23	2	1	61	0	0
97	20	23	2	1	62	0	0
98	21	18	2	1	56	0	0
99	21	18	2	1	55	0	0
100	21	21	1	1	0	0	0
101	21	22	2	1	64	0	0
102	21	15	2	1	46	0	0
103	21	15	2	1	45	0	0
104	22	21	2	1	64	0	0

105	22	17	2	1	52	0	0
106	22	22	1	1	0	0	0
107	23	20	2	1	42	0	0
108	23	23	3	1	-62	-66	23
109	23	23	3	1	-62	-61	23
110	23	20	2	1	61	0	0
111	23	23	3	1	-61	-36	23
112	23	13	2	1	36	0	0
113	23	23	3	1	-36	-33	23
114	23	12	2	1	33	0	0
115	23	23	3	1	-33	-66	23
116	23	23	1	1	0	0	0
117	24	15	2	1	42	0	0
118	24	3	2	1	12	0	0
119	15	24	2	1	42	0	0

Tab. 1: Breaker Table Matrix, IEEE Reliability Test System 1996

A.2.

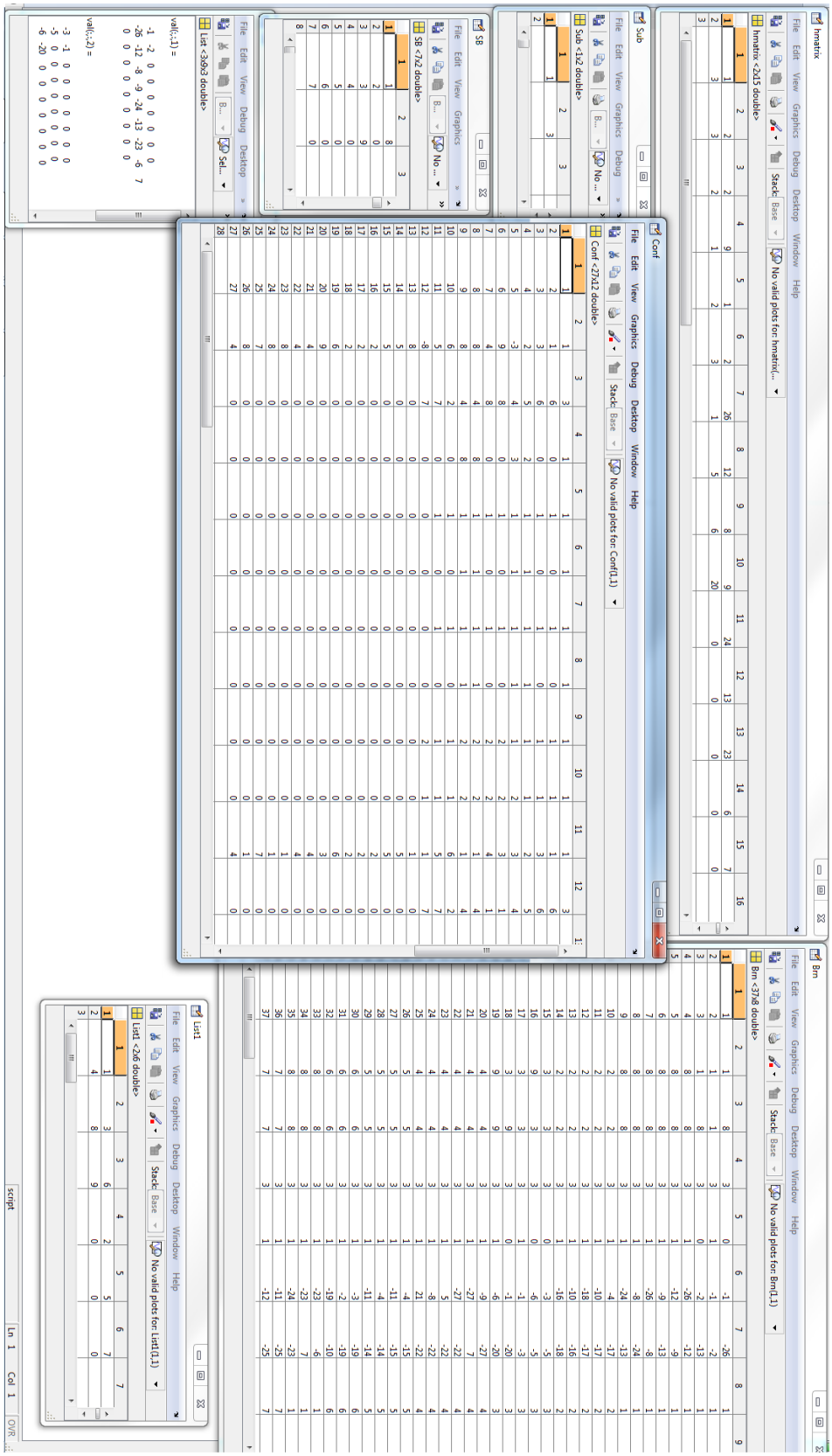
Circuit Number	Near Station	Far Station	Near Station[7]	Near Meas. Status	Far Meas. Status	Load Flow Avai.	Inf.	Near Station Area	Far Station Area	Orig. Near Station	Orig. Far Station
1	1	1	0	1	1	1	1	1	1	1	1
2	1	1	0	1	0	1	0	1	1	1	1
3	1	2	1	1	1	1	1	1	1	1	2
4	1	5	1	1	1	1	1	1	1	1	5
5	1	3	1	1	1	1	1	1	1	1	3
6	2	2	0	1	0	1	0	1	1	2	2
7	2	2	0	1	0	1	0	1	1	2	2
8	2	4	2	1	1	1	1	1	1	2	4
9	2	6	2	1	1	1	1	1	1	2	6
10	3	9	3	1	1	1	1	1	1	3	9
11	3	3	0	1	0	1	0	1	1	3	3
12	3	24	3	1	1	1	1	1	1	3	24
13	4	9	4	1	1	1	1	1	1	4	9
14	4	4	0	1	0	1	0	1	1	4	4
15	5	10	5	1	1	1	1	1	1	5	10
16	5	5	0	1	0	1	0	1	1	5	5
17	6	6	0	1	0	1	0	1	1	6	6
18	6	10	6	1	1	1	1	1	1	6	10
19	7	7	0	1	0	1	0	1	1	7	7
20	7	8	7	1	1	1	1	1	1	7	8
21	7	7	0	1	0	1	0	1	1	7	7
22	8	8	0	1	0	1	0	1	1	8	8
23	8	9	8	1	1	1	1	1	1	8	9
24	8	10	8	1	1	1	1	1	1	8	10
25	9	12	9	1	1	1	1	1	1	9	12
26	9	11	9	1	1	1	1	1	1	9	11
27	9	9	0	1	0	1	0	1	1	9	9
28	10	11	10	1	1	1	1	1	1	10	11
29	10	12	10	1	1	1	1	1	1	10	12
30	10	10	0	1	0	1	0	1	1	10	10
31	11	13	11	1	1	1	1	1	1	11	13
32	11	14	11	1	1	1	1	1	1	11	14
33	12	23	12	1	1	1	1	1	1	12	23
34	12	13	12	1	1	1	1	1	1	12	13
35	13	13	0	1	0	1	0	1	1	13	13
36	13	23	13	1	1	1	1	1	1	13	23

[7] Refer to section 5.3.3.

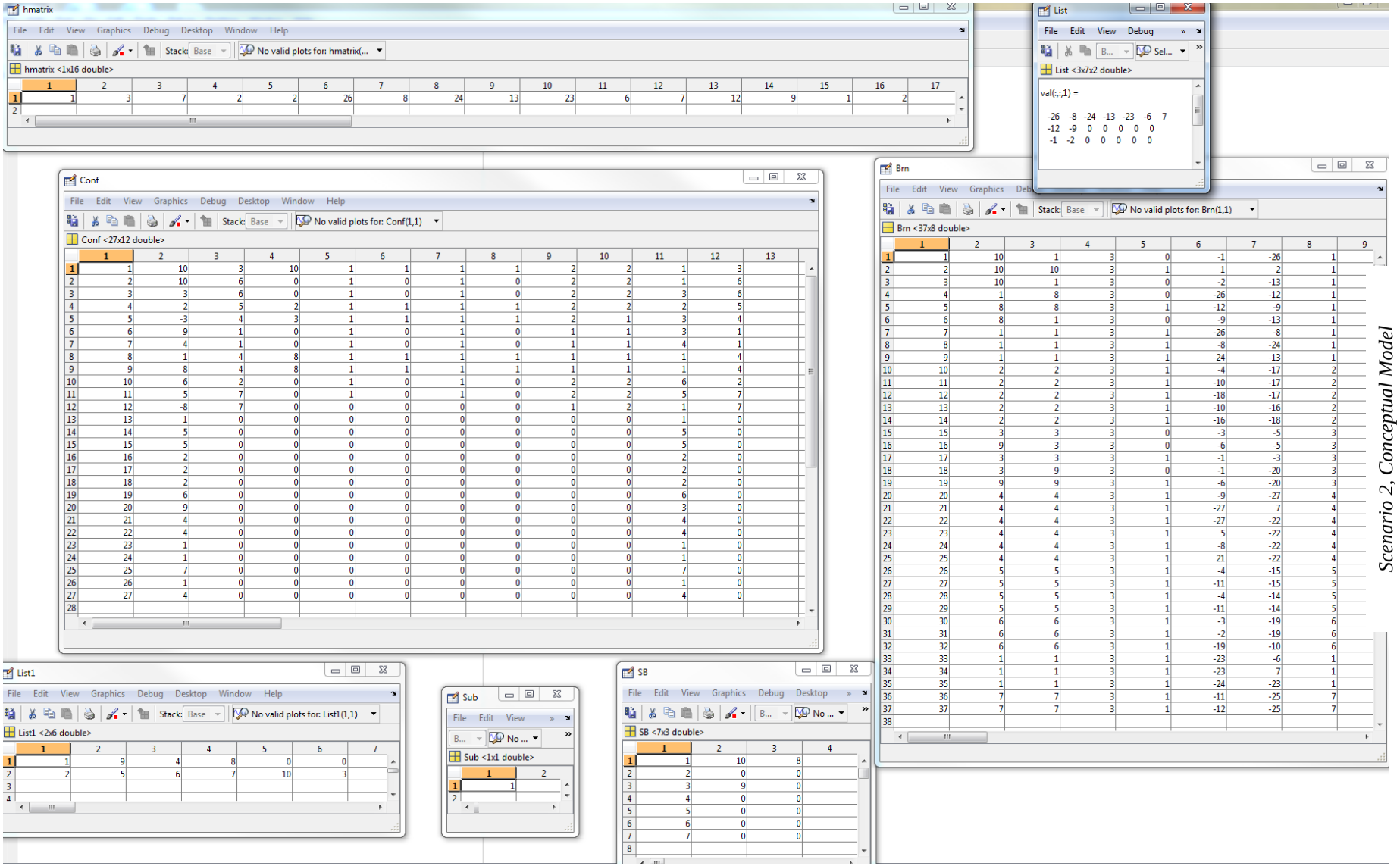
37	13	13	0	1	0	1	0	1	1	13	13
38	14	14	0	1	0	1	0	1	1	14	14
39	14	16	14	1	1	1	1	1	1	14	16
40	14	14	0	1	0	1	0	1	1	14	14
41	15	15	0	1	0	1	0	1	1	15	15
42	15	24	15	1	1	1	1	1	1	15	24
43	15	16	15	1	1	1	1	1	1	15	16
44	15	15	0	1	0	1	0	1	1	15	15
45	15	21	15	1	1	1	1	1	1	15	21
46	15	21	15	1	1	1	1	1	1	15	21
47	16	16	0	1	0	1	0	1	1	16	16
48	16	16	0	1	0	1	0	1	1	16	16
49	16	17	16	1	1	1	1	1	1	16	17
50	16	19	16	1	1	1	1	1	1	16	19
51	17	18	17	1	1	1	1	1	1	17	18
52	17	22	17	1	1	1	1	1	1	17	22
53	18	18	0	1	0	1	0	1	1	18	18
54	18	18	0	1	0	1	0	1	1	18	18
55	18	21	18	1	1	1	1	1	1	18	21
56	18	21	18	1	1	1	1	1	1	18	21
57	19	20	19	1	1	1	1	1	1	19	20
58	19	20	19	1	1	1	1	1	1	19	20
59	19	19	0	1	0	1	0	1	1	19	19
60	20	20	0	1	0	1	0	1	1	20	20
61	20	23	20	1	1	1	1	1	1	20	23
62	20	23	20	1	1	1	1	1	1	20	23
63	21	21	0	1	0	1	0	1	1	21	21
64	21	22	21	1	1	1	1	1	1	21	22
65	22	22	0	1	0	1	0	1	1	22	22
66	23	23	0	1	0	1	0	1	1	23	23
67	1	0	0	0	0	0	0	0	0	1	0
68	2	0	0	0	0	0	0	0	0	1	0
69	3	0	0	0	0	0	0	0	0	1	0
70	4	0	0	0	0	0	0	0	0	1	0
71	5	0	0	0	0	0	0	0	0	1	0
72	6	0	0	0	0	0	0	0	0	1	0
73	7	0	0	0	0	0	0	0	0	1	0
74	8	0	0	0	0	0	0	0	0	1	0
75	9	0	0	0	0	0	0	0	0	1	0
76	10	0	0	0	0	0	0	0	0	1	0
77	11	0	0	0	0	0	0	0	0	1	0
78	12	0	0	0	0	0	0	0	0	1	0
79	13	0	0	0	0	0	0	0	0	1	0
80	14	0	0	0	0	0	0	0	0	1	0
81	15	0	0	0	0	0	0	0	0	1	0
82	15	0	0	0	0	0	0	0	0	1	0
83	16	0	0	0	0	0	0	0	0	1	0
84	17	0	0	0	0	0	0	0	0	1	0
85	18	0	0	0	0	0	0	0	0	1	0
86	19	0	0	0	0	0	0	0	0	1	0
87	20	0	0	0	0	0	0	0	0	1	0
88	21	0	0	0	0	0	0	0	0	1	0
89	22	0	0	0	0	0	0	0	0	1	0
90	24	0	0	0	0	0	0	0	0	1	0

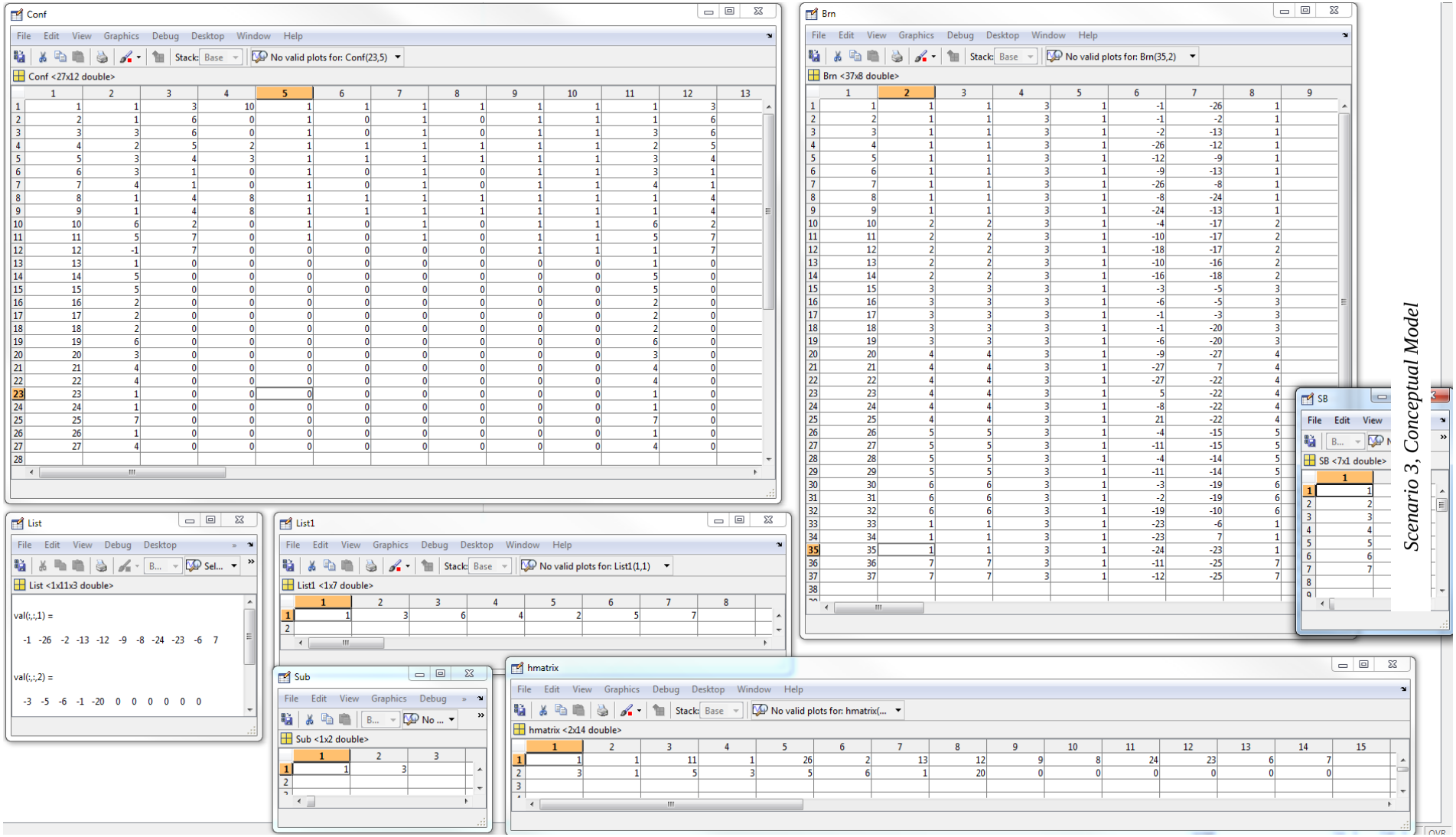
Tab. 2: Configuration Matrix, IEEE Reliability Test System 1996

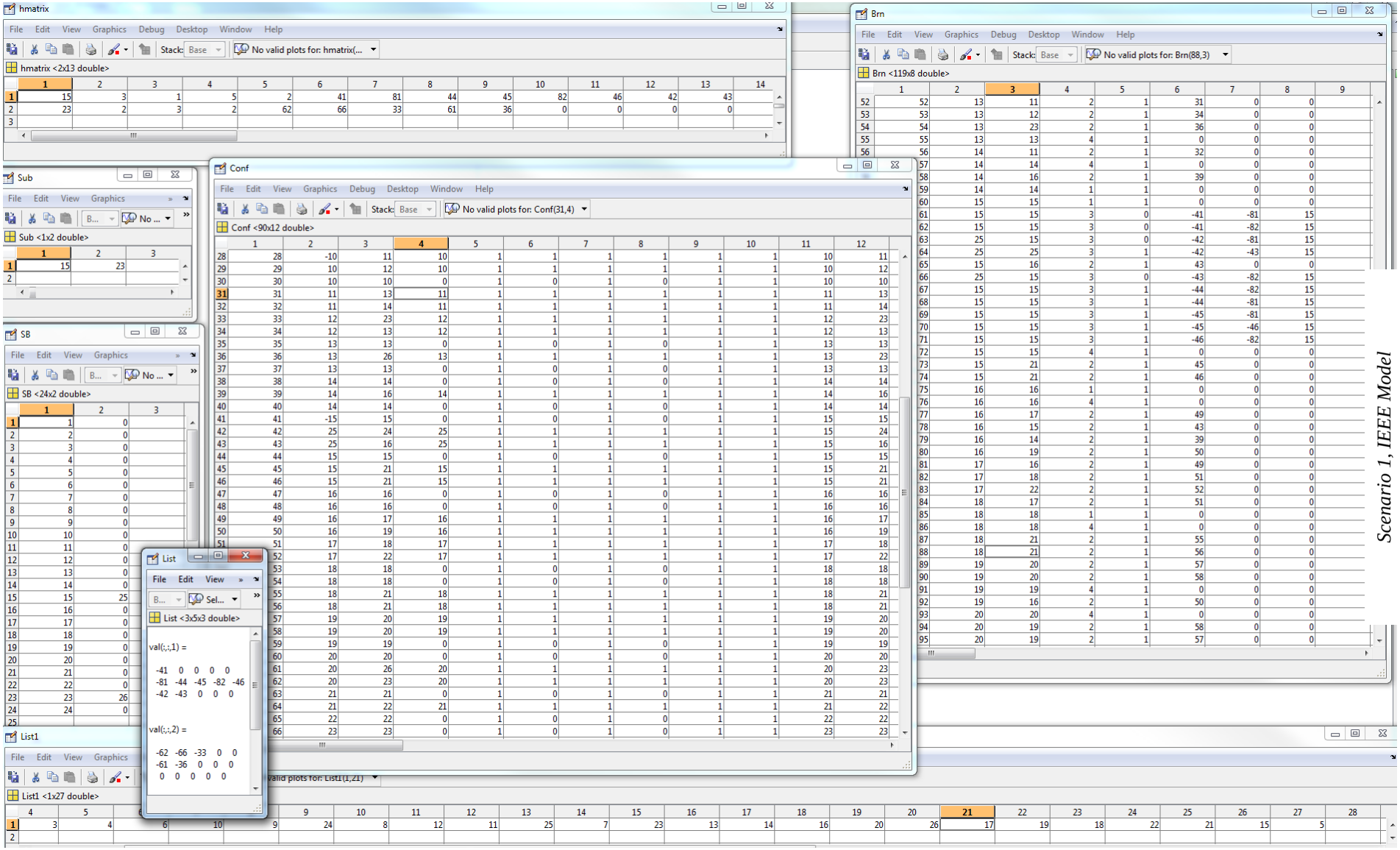
A.3.



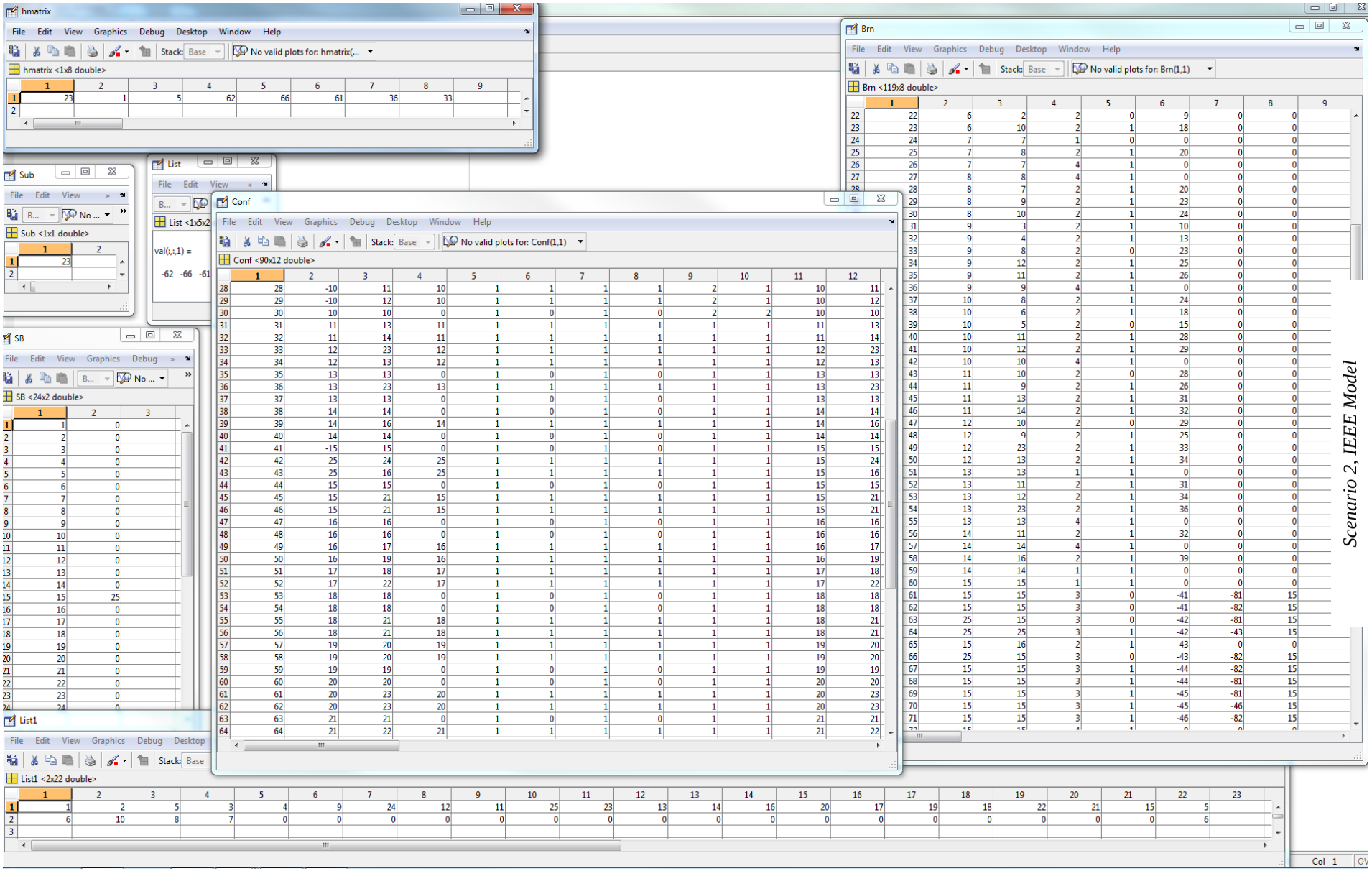
Scenario 1, Conceptual Model

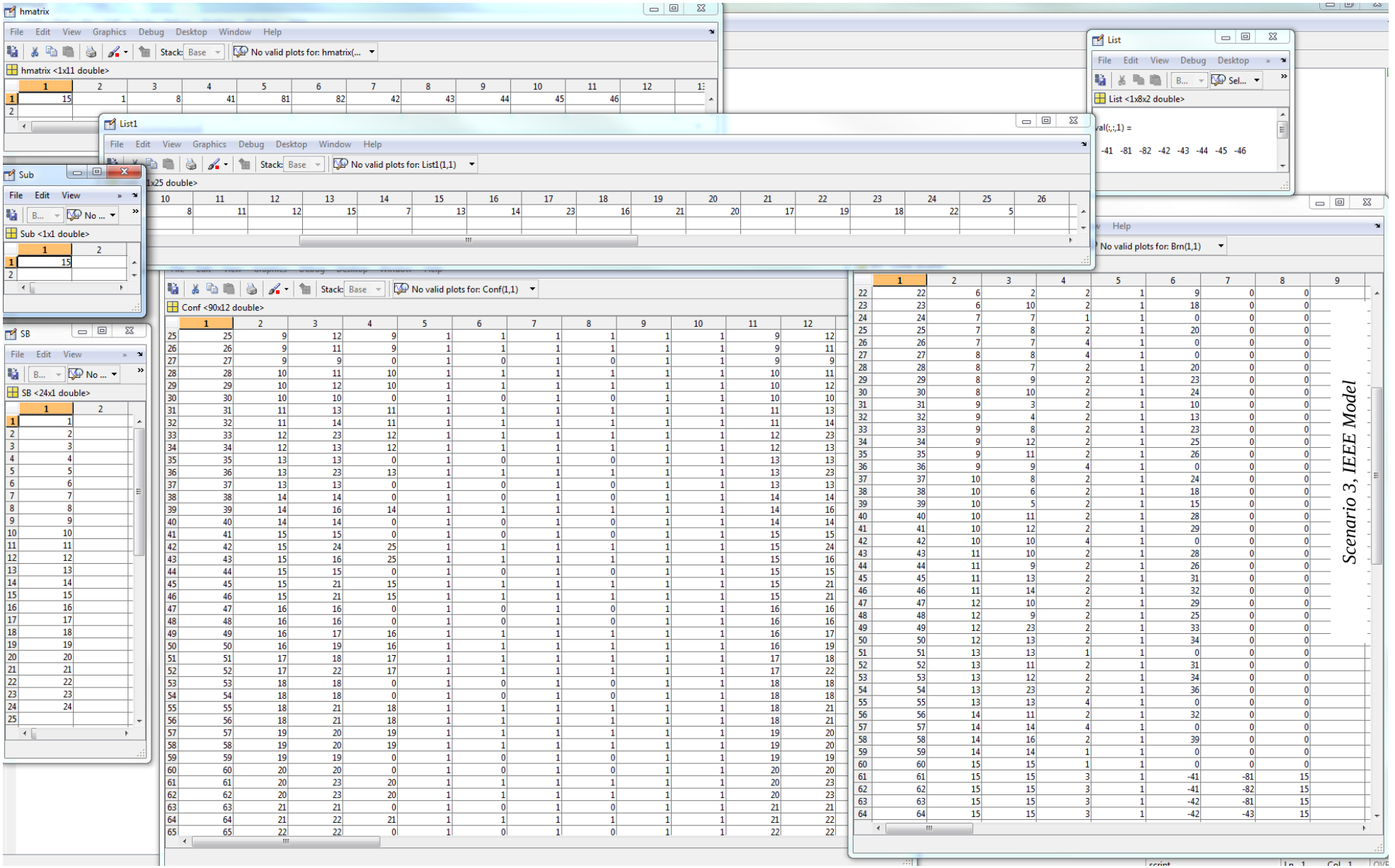

hmatrix
hmatrix <1x16 double>
List
List <3x7x2 double>
val(:,:,1) =
-26 -8 -24 -13 -23 -6 7
-12 -9 0 0 0 0 0
-1 -2 0 0 0 0 0
Conf
Conf <27x12 double>
Brn
Brn <37x8 double>
List1
List1 <2x6 double>
Sub
Sub <1x1 double>
SB
SB <7x3 double>
Scenario 2, Conceptual Model


Conf
Conf <27x12 double>
Brn
Brn <37x8 double>
SB
SB <7x1 double>
List
List <1x11x3 double>
val(:,:,1) =
-1 -26 -2 -13 -12 -9 -8 -24 -23 -6 7
val(:,:,2) =
-3 -5 -6 -1 -20 0 0 0 0 0 0
List1
List1 <1x7 double>
Sub
Sub <1x2 double>
hmatrix
hmatrix <2x14 double>
Scenario 3, Conceptual Model



Scenario 1, IEEE Model


hmatrix
hmatrix <1x8 double>
Sub
Sub <1x1 double>
List
List <1x5x2
SB
SB <24x2 double>
Conf
Conf <90x12 double>
Brn
Brn <119x8 double>
List1
List1 <2x22 double>
Scenario 2, IEEE Model


hmatrix
hmatrix <1x11 double>
List1
1x25 double>
Sub
Sub <1x1 double>
SB
SB <24x1 double>
Conf <90x12 double>
List
List <1x8x2 double>
Brn(1,1)
Scenario 3, IEEE Model