Electric Power Systems Research 106 (2014) 188–202

Contents lists available at ScienceDirect

Electric Power Systems Research

journal homepage: www.elsevier.com/locate/epsr

An efficient automated topology processor for state estimation of power transmission networks

Mostafa Farrokhabadi[a], Luigi Vanfretti[a,b,*]

[a] KTH Royal Institute of Technology, School of Electrical Engineering, Electric Power Systems Department, Teknikringen 33, SE 100 44 Stockholm, Sweden
[b] Statnett SF, Research and Development, Nydalen Allé 33 0484 Oslo, Norway

ARTICLE INFO

Article history:
Received 23 April 2012
Received in revised form 13 May 2013
Accepted 22 August 2013
Available online 27 September 2013

Keywords:
Topology processor
State estimation
Real-time simulation
Phasor measurement unit

ABSTRACT

A robust network topology processor that can be utilized in both traditional and PMU-based state estimators is developed. Previous works in the field of topology processing are scrutinized and their drawbacks are identified. Building on top of the state of the art, an algorithm covering the limitations of available topology processing approaches and including new features is proposed. The presented algorithm was implemented in MATLAB and tested using two different power networks with detailed substation configurations (bus/breaker models) including a modified version of the IEEE Reliability Test System 1996. As the topology processor is intended to supply network topologies to a PMU-based Sate Estimator, the IEEE Reliability Test System 1996 is simulated in real-time using the eMegaSim Opal-RT real-time simulator which is part of “SmarTS Lab” at KTH Royal Institute of Technology. Testing is carried out through several test scenarios and computation times are calculated. It is shown that the computation times are adequate for supporting a PMU-only state estimator.

1. Topology processing

1.1. Synopsis

State estimation is a key energy management system (EMS) function that provides static estimates of the system states, i.e., bus voltages and angles, and line active and reactive power flows [1]. Topology processing is a key step in the state estimation process. It refers to the determination of the system topology through available measurements and connectivity data of switch status indicators and its link to a static database; this database holds relevant information about the parameters of such a topology.

At the first glance, the task seems to be straightforward: an open switch indicates that a line is disconnected. However, the problem is more complex in practice. Considering a real power system, there are numerous substations with different configurations, and various interconnections. On top of that, there are moments when an interconnected power system is split into separate islands due to a specific switching pattern. The task of a network topology processor is to correctly deal with all of these complexities.

The objective of this paper is to develop a robust topology processor capable of supporting both traditional SEs and fast PMU-only state estimators [2,3]. The algorithm is documented in detail and relevant testing has been performed using two different test systems with detailed substation configurations, including a fictitious network and a modified version of IEEE Reliability Test System 1996 [4]. The latter is simulated in real-time using the eMegaSim Opal-RT real-time simulator [5].

From the practical point of view, several steps are involved in a typical topology processor [6]. The first step is to accurately process input data. This data consists of pre-defined constant connectivity parameters and switch statuses. Traditionally, data is telemetered through analog transmitters using TCP/IP over the ICCP protocol. However, the arrival of PMUs made it possible to send the switches status digitally with a much higher rate compared to conventional telemetry methods. These digital data is included in the IEEE C37.118-2011 protocol [7]. It is shown in this paper that the performance of the proposed algorithm is adequate for its use with high rate PMU data as input.

The second step in topology processing is to analyze different substation configurations. In fact, due to various substation configurations, an open breaker does not necessarily disconnect a line from the substation. In addition, there might be moments that one single substation is split into two (or more) nodes. Similarly, there may be cases in which these previously split nodes merge back together to reform the original station.

---

* Corresponding author. Tel.: +46 87906625; fax: +46 87906510.
E-mail address: luigiv@kth.se (L. Vanfretti).

0378-7796/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.epsr.2013.08.014

The next major step carried out by a topology processing engine is to detect if islanding has occurred in a power network. Reversely, it should be able to detect if some separated islands have been unified. Identification of energized and de-energized islands is also a duty of a practical topology processor. In an energized island, there is at least one operating generator. This task is also of high importance, as the de-energized islands should not participate in the state estimation process. Therefore there is no need to formulate the equations related to them.

The openly available works on the implementation of the current topology processors pose several challenges for a practical implementation; this is mostly due to a lack of rigor in the description and documentation of these methods.

To have a proper background that motivates the development of a new algorithm, the most relevant previous works are discussed below. Please note that the works considered here are related to the term "Topology Processing" and not "Topology Estimation". The former is designated to the action of determining the topology of a power system from input data (breakers' status, and bus connectivity data). Topology estimation [13] refers to the action of estimating the topology of a power system. To carry out this, the topology of the system has already been determined a priory by a topology processor. However, this topology might have errors (due to bad input data). Thus, the process of topology estimation aims to detect these errors and mitigate them. The result is a sanitized topology of the network. This article does not propose algorithms for topology estimation.

1.2. *Literature review*

1.2.1. *Automatic power system network topology determination [8]*

Introduced by Sasson *et al.*, this is the cornerstone work in the area and the common reference for the majority of proposed methods. Similarly, the method proposed in this article builds on top of the algorithm in Ref. [8]. However, we identified some of the drawbacks that limit its implementation and performance.

First, the algorithm is communicated in natural language form (written English); however, there are other dialects (such as flowcharts, influence diagrams) that are better suited for a translation into a software implementation. No flowcharts or mathematical descriptions are provided in Ref. [8]. This is challenging because for an implementation into actual computer code a more detailed functional specification and a step-by-step flowchart is needed. This flowchart must clearly describe in detail the steps needed to derive code for actual execution; it should be very similar to "data flow diagrams" which are used in computer programming (this is what we have attempted in our manuscript, as shown in the forthcoming sections).

There are other limitations in this algorithm that affect its performance. Whenever a breaker status change occurs, the algorithm needs to perform topology processing for all the substations with breaker statuses different from their initial values. A more sensible approach is to compare the breakers statuses with the ones of the most recent snapshot. This inability is due to the fact that the TP does not track the changes that have occurred in the previous cycle; this is a consequence from the lack of a number assignation procedure to mark splitting and merging nodes. As a result the TP has to redefine the whole topology of the system in every snapshot so to avoid numbering uncertainty. Even if the algorithm processes the topology and compares it to a pre-defined initial topology, it may significantly change the numbering of the nodes which were previously involved in splitting/merging incidents.

Another difficulty takes place during node merging/splitting. As the information regarding the original station numbers are not saved in the "configuration matrix" the algorithm may malfunction because this matrix is used in islanding analysis.

Additionally, the algorithm only considers changes within a substation. Switching changes that are not located within the substation configuration such as those placed on the lines are not accounted for. Furthermore, the rules of this TP are not able to account for ring-bus with series breakers configurations.

Finally, the TP is not able to detect if the islands are energized or de-energized. This is very important as the parts of the network which are not energized should not participate in state estimation or static analysis (power flow solution).

1.2.2. *Real-time modeling of power networks [9]*

Real-time modeling of power networks in [9] refers to a set of computer-based functions that aims to represent the current operation states of the power network. It comprises different functions such as topology processing, observability analysis, state estimation, bad data detection and external network modeling. In Ref. [9] all the required steps for real-time modeling of a power network are explained and general procedures for each one is described. The discussion on topology processing in this article is a short general definition of the topology-processing concept. The necessary steps for TP are only communicated in natural language form (written English) and no detailed dialects (such as flowcharts) are provided for computer software implementation.

1.2.3. *A topology processor that tracks network modifications over time [10]*

Ref. [10] provides an approach to save the outputs of the topology engine for each execution cycle; it uses this saved data to reduce the calculation burden of the state estimation process. Although the paper has a top-level descriptive flowchart, it does not provide a particular algorithm for the topology engine itself. The methodology used in the topology engine in Ref. [10] is based on the one proposed in Ref. [8]. Therefore, it is exposed to the same drawbacks as discussed in Section 1.2.1.

1.2.4. *A new algorithm of topology analysis based on PMU information [11]*

To the knowledge of the authors, this work is the only one that considers the usage of PMUs for topology processing. The basic idea of this work is to measure the current flows and compare them with a pre-defined threshold. However, necessary elements for topology processing, such as how to deal with different configurations, automatic numbering, and islanding checks, are not addressed by the algorithm proposed. Furthermore, the algorithm neglects the fact that when a change occurs in the network, this change may affect large portion of the network (i.e. this effect is not strictly localized). For example, if two lines are connected in series, the disconnection of one may result in the current flow through the other line to be low. This issue is investigated in detail in Ref. [14] and possible remedies are provided.

1.2.5. *A new approach to initializing and updating the topology of an electrical network [12]*

The method introduced in Ref. [12] uses graph theory. The power system network is transformed to a graph, and 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.

The TP documented in Ref. [12] is more of an industrial report, with limited information on the dialectics for its implementation. Specifically, the methodology to solve the sparse matrix and determine the network topology is not elaborated. Furthermore, the explanations on how to transforms the power network to an equivalent graph is explained using an example, and no general