SPECIFICATION

1) INTRODUCTION

Need for explicitly modeling physical equipment and devices has resulted in making node/breaker network model. This model is mostly used for controlling and monitoring power networks. Majority of electrical companies in the world were using their own ways of describing network elements so it was important to come up with standardized model, CIM (*Common Information Model*), which everyone can use to easily exchange information. CIM is object oriented standard which represents node/breaker model in detail. Additionally, for making some optimizations and calculations to network, it is necessary to have network topology. Network topology can be determined from statuses of switches (breakers and disconnectors) and their correlation with elements in network, which can be found in CIM database.
Switch is not an element on which one can do some calculations, so for advanced network numerical analysis (egz. state estimation, optimal power flow,..) it is better to use an abstract model known as bus/branch network model. This model eliminates switches based on their open/close status and groups connectivity nodes, connected with closed switches, into one topological node (bus). Other elements of electrical network are modeled as topological branches and this process of conversion is called network topology processing. Because normal (default) statuses of switches aren't always true to current situation, it is important to keep representation of each element from the node/breaker model for directly applying dynamic changes. Once the bus/branch model is built, there is no need to rebuild the whole model again after a change occurs. Only the affected entities have to be altered.

2) DESCRIPTION OF ELEMENTS IN CIM MODEL

In CIM model, equipment is a physical device in the power system and conducting equipment is equipment designed to conduct electricity. Conducting equipment is only connected through terminals and connectivity nodes, where terminal is the point at which connections can be made to a network and connectivity node is a point of zero impedance for connecting terminals. Equipment can have one or more terminals, terminal is connected to one connectivity node, and one connectivity node has one or more terminals connected on them.

For test-case, ODS Koprivnica model is used. *Physical elements* in CIM model from Koprivnica are: Switch, BusBarSegment, ACLineSegment, EquivalentInjection and EnergyConsumer. Other elements in CIM model are: ConnectivityNode and Terminal.

Classes in CIM have multiple attributes, but for the first step in processing only the attribute for mRID of element is needed.

Elements:

- ***Switch***: has 2 associated terminals and subclasses Breaker and Disconnector. Status of switch needs to be externally retrieved from SCADA
- ***BusBarSegment***: has 1 associated terminal
- ***ACLineSegment***: has 2 associated terminals and parameters for resistance (r), inductance (x), shunt susceptance (bch), shunt conductance (gch) and conductor length

- ***EquivalentInjection***: has 1 associated terminal. Exact power data need to be externally retrieved from SCADA
- ***EnergyConsumer***: has 1 associated terminal. Exact power data need to be externally retrieved from SCADA
- ***ConnectivityNode***: has one or more associated terminals
- ***Terminal***: has reference for belonging ConnectivityNode and ConductingEquipment, plus ordinal number of terminal for belonging physical element

3) TOPOLOGICAL PROCESSING

Switch is closed if its status is equal to 1, otherwise it is open. If there is a closed switch between two connectivity nodes cn1, cn2 and, for egz., cn1 < cn2 (where "<" is some defined operator for comparing elements), then cn1 becomes cn2. This process is called merging of nodes (into numerically or alphabetically higher one).

Input variables for topological processing:

- ***node set***: sorted mRIDs of all connectivity nodes
- ***asset map***: mRID of element is key for instance of that element
- ***switch map***: mRID of switch is key for externally retrieved status of that switch

Additionally made map in function:

- ***terminal map***: mRID of physical element is key for list of mRIDs of associated terminals. Initialization of map is based on *ConductingEquipment* element referenced from element *Terminal*. From *asset map* all *Terminal* elements which have same mRID of reference for *ConductingEquipment* as key-mRID need to be found. mRIDs of these *Terminal* elements are put in a list.

To bypass that ConnectivityNode of every ACLineSegment becomes bus, it is necessary to connect all segments of line into one.
Additionally made class in function:

- ***ACLine***: has 2 associated terminals and parameters for resistance (r), inductance (x), shunt susceptance (bch), shunt conductance (gch) and conductor length. It also holds an attribute *number* for how many segments it is made of. For initialization set all values to 0.

Algorithm for making an instances of *ACLine* class:
For every element in *asset map* (iterate through keys, assuming they are sorted)**:**
  If it is *ACLineSegment*:
    Initialize *ACLine*, mRID is set to mRID of current *ACLineSegment*
    From *terminal map*, first associated terminal of current *ACLineSegment* is final first terminal of *ACLine*, and second terminal is trial second terminal
    In first associated *Terminal*, change reference for *ConductingEquipment* from current *ACLineSegment* to new *ACLine* (ordinal number stays the same)
    ***number* is set to 1**
    Attributes r,x,bch,gch and conductor length of *ACLine* are set to the same attributes of *ACLineSegment*
    From *node set* remove *ConnectivityNode* of second terminal (from reference of *Terminal* element find associated *ConnectivityNode*)
    For this element in *asset map* change its value to ACLine, and go to next element

While next element in *asset map* is *ACLineSegment*:

From *terminal map*, second associated terminal of current *ACLineSegment* is trial second terminal of *ACLine*

Attributes r,x,bch,gch and conductor length of *ACLineSegment* are summed up respectively with attributes of *ACLine*

***number* is increased by 1**

From *node set* remove *ConnectivityNode* of both terminals (from reference of *Terminal* element find associated *ConnectivityNode*)

From *asset map* delete key for this *ACLineSegment* and go to next element in map

From *terminal map* delete key for this *ACLineSegment*

/**Now we have the last segment of line**/

From *terminal map*, second associated terminal of current *ACLineSegment* is final second terminal of *ACLine*

Attributes r,x,bch,gch and conductor length of *ACLineSegment* are summed up respectively with attributes of *ACLine*

***number* is increased by 1**

From *node set* remove *ConnectivityNode* of first terminal (from reference of *Terminal* element find associated *ConnectivityNode*)

From *asset map* delete key for this *ACLineSegment*

From *terminal map* delete key for this *ACLineSegment*

Using mRID of *ACLine* as key in *asset map,* replace value with instance of this *ACLine* (from first segment of line to whole line)

Using mRID of *ACLine* as key in *terminal map,* replace value with new list which consist of first and second terminal of this ACLine

For second associated *Terminal* of *ACLine*, change its reference from *ACLineSegment* to *ACLine* (ordinal number stays the same)

/**Whole line is now connected**/

Two more additionally made maps in function:

- ***merge map***: mRID of connectivity node is key for mRID of connectivity node into which it is merged. Initialization of map is map[key]=key, where keys come from *node set*
- ***connectivity map:*** mRID of connectivity node is key for list of mRIDs of associated terminals. Initialization of map is based on *ConnectivityNode* element referenced from element *Terminal*. From *asset map* all *Terminal* elements which have same mRID of reference for *ConnectivityNode* as key-mRID need to be found. mRIDs of these *Terminal* elements are put in a list.

Algorithm for TP:

For every node in *node set*

Get list of terminals from *connectivity map* where key is mRID of current node

For every terminal in list

If it has connected closed switch (from *asset map* get instance of Terminal element and check if its reference for ConductingEquipment is switch; if it is, use mRID of that switch as key in *switch map* and check if its status is equal 1)

Get the other terminal of that switch (from *terminal map* get list of two terminals and choose the other one)

Save mRID of associated node for the other terminal (from *asset map* get instance of Terminal element and its reference for belonging ConnectivityNode)

If there were closed switches, find max of saved mRIDs

For current and saved nodes, except one with max mRID, in *merge map* rewrite value with max mRID

Transfer all terminals from nodes with smaller mRIDs to node with max mRID (in *connectivity map* for saved nodes with smaller mRID move values to value of max key-mRID)

/**
Because some nodes that represented maximum in one iteration, could represent minimum in some of next iterations, it is necessary to iterate through *node set* in reversible way and make adjustments to *merge map*. *Connectivity map* is updated.
**/

For every node in *node set* (in reversible way)

Merge node to associated maximum node (merge map[mRID] = merge map[merge map [mRID]])

/**
If for a node is valid: *merge map[mRID] = mRID,* it is called *final node*. TopologicalNode is a set of connectivity nodes which are merged into same ConnectivityNode.
– izgled klase ce biti preuzet iz talijanske baze
**/

For every key-node in *merge map*

Put it in set of associated TopologicalNode (if TopologicalNode doesn't exist yet, make new and add it to his set)

Output of topological processor are topological nodes (sorted in a set?).

4) PARAMETERS PROCESSING

Most of parameters from ATTEST ICT database don't match parameters in CIM database, so they need to be processed first.

4.1) Nodal admittance matrix

Voltage magnitude (per unit), voltage angle, real power demand (MW) and reactive power demand (MWAr) are results of State Estimation (SE). Input data for SE is nodal admittance matrix (Ybus). Ybus is a *N x N* matrix describing a linear power system with N buses. It represents the nodal admittance of the buses in a power system.

Additionally made map in function:

- *final map*: mRID of final node is key for increasing number, starting from 0.

Algorithm for finding who is connected to which bus:

Set dimension N of matrix to zero

For every key in *merge map*:

If node is final node:

mRID of node is key in *final map* for value N

N is increased by one

/**Now we know number of buses in the model**/

Matrix with dimension N is set to zero

For every node in *merge map*:

If node is *final node*:

Go trough *connectivity list* of that node:

If reference for ConductingEquipment of *Terminal* is *ACLine*:

From *terminal map* get the other *Terminal* of *ACLine* (where key is mRID of ACLine)

From other *Terminal* take reference for *ConnectivityNode*

For found *ConnectivityNode*, from *merge map* find *TopologicalNode* to which it belongs (key is mRID of found ConnectivityNode)

Calculate admittance (Using attributes r, x from *ACLine*: admittance = r/(r^2+x^2) - j*[x/(r^2+x^2)])

On place (final map[mRID of first node], final map[mRID of second node]) **in matrix set value to: admittance*(-1)**

On place (final map[mRID of first node], final map[mRID of first node]) **in matrix add up value of admittance**

On place (final map[mRID of first node], final map[mRID of first node]) **in matrix add up shunt value** (Using attributes gch, bch from *ACLine*: shunt = gch/2 + j*(bch/2))

Nodal admittance matrix is made and can be used in state estimation.

4.2) Converting values to "*per-unit*"

Calculations are simplified because quantities expressed as per-unit don't change when they are referred from one side of a transformer to the other. All quantities are specified as multiples of selected base values (p.u. = actual value / base).

Voltage and power base are known (?).
Impedance base is calculated with formula: Z base = (V base)^2 / S base [(kV)^2 / MVA].
Actual value of impedance is taken from r,x attributes of ACLine: Z actual = r + j*x.
Impedance per unit is then: Z p.u. = Z actual / Z base.

Notice that Z base is real number, so Z p.u. is again complex number where real part represents resistance p.u., and complex part represents reactance p.u.. These two values are one of the attributes for branch in ATTEST ICT database.

https://cimug.ucaiug.org/Meetings/Europe2015/Shared%20Documents/2015%20CIM%20Gdansk%20Meeting%20-%20MKCelik%20Nexant%20Final.pdf

https://en.wikipedia.org/wiki/Terminal_(electronics)
https://en.wikipedia.org/wiki/Nodal_admittance_matrix
https://en.wikipedia.org/wiki/Per-unit_system