Modeling of Remedial Action Schemes and Relays in Power Flow Simulations

Jamie Weber, Ph.D.
Director of Software Development
weber@powerworld.com
217-384-6330 ext. 13

PowerWorld Corporation

Introduction

- James Weber (Jamie)
  - Director of Software Development at PowerWorld Corporation since 1997
- Experience with RAS and Relays
  - For 15 years we have been working with engineers on directly modeling the logic and actions that describe RAS and Relays in software
  - We have a lot of experience looking at descriptions of RAS and encoding them in software
  - We have a lot of experience adding new features to software to permit additional wrinkles in how RAS is defined.
  - This feature set has evolved incrementally over the past 15 years doing several dozen very small projects to enhance the software feature set

Purpose of Presentation

- Investigate your RAS and Relaying
  - Find who owns this information in your company
  - Details matter
- Introduce a structure and text file format for defining your RAS (we'll do this through examples)
  - http://www.powerworld.com/files/PowerWorld_RASFileFormat.pdf
- Encourage attendance at In-Person Workshop at WECC offices in Salt Lake City on May 27 – 29, 2015
  - Starts 1 PM on May 27, Ends at Noon on May 29
  - There will be no WebEx for this meeting. In-Person Only.

Monday May 25	Tuesday May 26	Wednesday May 27	Thursday May 28	Friday May 29
		Classroom Style	**Break-Out Groups**	
		1 PM		**Noon**

Classroom Style and Break-Out Groups

- Classroom Style Portion
  - Presentations on how to implement the various input parts that represent RAS and Relay models
    - Contingency Actions
    - Boolean Logic
    - Lookup tables and Expressions
  - Bring your laptop and work along with us
- Break-out Groups Portion
  - There will be several engineers from throughout WECC with experience implementing their RAS and Relay Models working attending this workshop as well
  - We will break the attendees into smaller groups to spend time implementing their actual RAS
  - *Bring your RAS descriptions* so you can work on them
  - Bring your laptop and go home with some real RAS modeled
- Entire workshop is hands-on → There will be no WebEx/Phone

Overview of Presentation

- Define RAS and Relaying
- Contingency Definitions
- Legacy Methods for Modeling RAS and Relays
- Who has *details* on RAS modeling
- Describe parts of RAS modeling
- Two detailed examples of RAS and how they are implemented in software

What is RAS?

- NERC wrote a document to define this in June 2014
  - http://www.nerc.com/pa/Stand/Prjct201005_2SpclPrtctnSstmPhs2/FAQ_RAS_Definition_0604_final.pdf
  - WECC was well represented on this team

Project 2010-05.2 – Special Protection Systems SDT		
	Participant	**Entity**
Chair	Gene Henneberg	NV Energy / Berkshire Hathaway Energy
Vice Chair	Bobby Jones	Southern Company
Member	Amos Ang	Southern California Edison
	John Ciufo	Hydro One Inc.
	Alan Engelmann	ComEd / Exelon
	Davis Erwin	Pacific Gas and Electric
	Sharma Kolluri	Entergy
	Charles-Eric Langlois	Hydro-Quebec TransEnergie
	Robert J. O'Keefe	American Electric Power
	Hari Singh	Xcel Energy
NERC Staff	Al McMeekin (Standards Developer)	NERC
	Erika Chanzes (Standards Developer)	NERC
	Phil Tatro (Technical Advisor)	NERC
	Bill Edwards (Legal Counsel)	NERC

NERC Definition: **Protection System**

- The NERC Glossary of Terms defines a Protection System as
  - Protective relays which respond to electrical quantities
  - Communications systems necessary for correct operation of protective functions
  - Voltage and current sensing devices providing inputs to protective relays
  - Station dc supply associated with protective functions (including batteries, battery chargers, and non-battery-based dc supply)
  - Control circuitry associated with protective functions through the trip coil(s) of the circuit breakers or other interrupting devices.

NERC Definition:
RAS (Remedial Action Scheme)

- A scheme designed to detect predetermined System conditions and automatically take corrective actions that may include, but are not limited to, curtailing or tripping generation or other sources, curtailing or tripping load, or reconfiguring a System(s). RAS accomplish one or more of the following objectives:
  - Meet requirements identified in the NERC Reliability Standards
  - Maintain System stability
  - Maintain acceptable System voltages
  - Maintain acceptable power flows
  - Limit the impact of Cascading
  - Address other Bulk Electric System (BES) reliability concerns.
- These schemes are not Protection Systems; however, they may share components with Protection Systems.

Note: Most of this document is dedicated exceptions of what is NOT RAS

Note: Term “RAS” is used instead of “SPS” (Special Protection Scheme) to avoid confusion

My Simple Definition

- Protection System Device (typically a Relay)
  - Device monitors a small set of mostly local signals
  - Usually protects a single piece of equipment from damage
  - Actions are typically limited to opening or closing breakers
- Remedial Action Scheme (RAS)
  - Control scheme monitors a larger set of signals (potentially more than local signals)
  - Protects one or more pieces of equipment or prevents larger system-wide or region-wide collapse
  - Actions are more diverse

What is the Time-Frame?

- These definitions make no distinction about the time-frame of the Relay or RAS actions
  - If milliseconds to a few seconds → a Transient Stability model is necessary
  - If tens of seconds to minutes with automated response of Relay or RAS → Power Flow solution simulation only
  - Minutes of response as a person (the operator) walks over and takes off the shelf the binder entitled "What to do when stuff happens" → again Power Flow

- Majority of this webinar and the workshop in May focuses only on topics related to the steady state power flow solutions

What is Measured

- System Protection definition says "*Voltage and current sensing devices providing inputs to protective relays*"
  - Voltage and current are available in a Power Flow solution
  - But time-frame again may require Transient Stability
- RAS definition says "*scheme designed to detect predetermined System conditions*"
  - This may be voltage and current → Power Flow
  - But could be generator speed, frequency, etc. → Transient Stability

Accept Questions

- Pause for questions from audience

In software: when are RAS and Relays used?

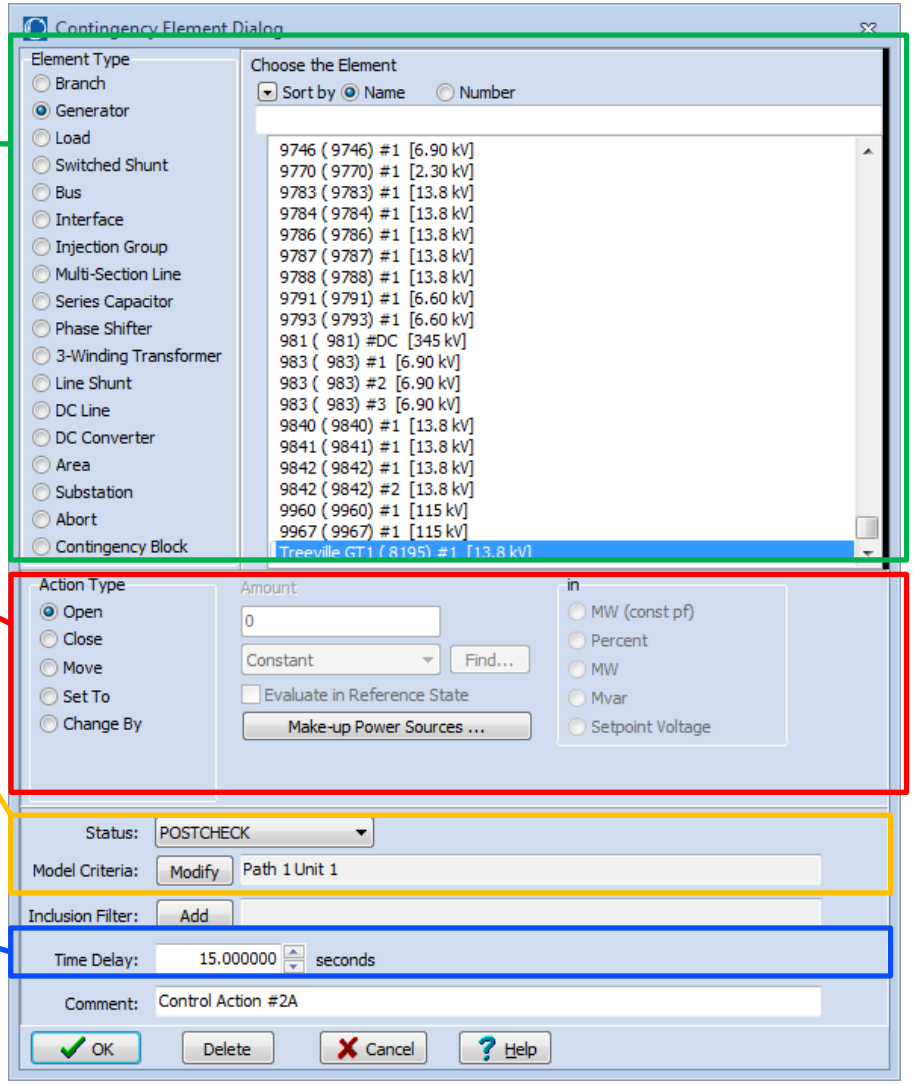
- Clearly *not* under normal operating conditions
  - When solving a base case power flow solution → RAS should not be doing anything
- Within a software tool, when are RAS and Relays going to matter?
  - **Contingency, Contingency, Contingency**
  - RAS will respond to the changes that occur during an unexpected event → **Contingency**
- For RAS and Relay models to be useful → first obtain or define a list of contingencies

Defining Contingencies

- Each **Contingency** has *Name* which is a unique string used to identify it
  - We need the name because this will also be used to identify it when looking at results
- Each **Contingency** is then made up of many **ContingencyElements**
- A **ContingencyElement** describes
  - *Object* to which an action is applied
  - *Action* that occurs
  - *Criteria, CriteriaStatus, TimeDelay* under which the action occurs (Boolean logic, where to apply, ordering)
    - These aren't needed for plain contingency definitions, but will become vital in the RAS modeling which will be discussed in examples

ContingencyElement

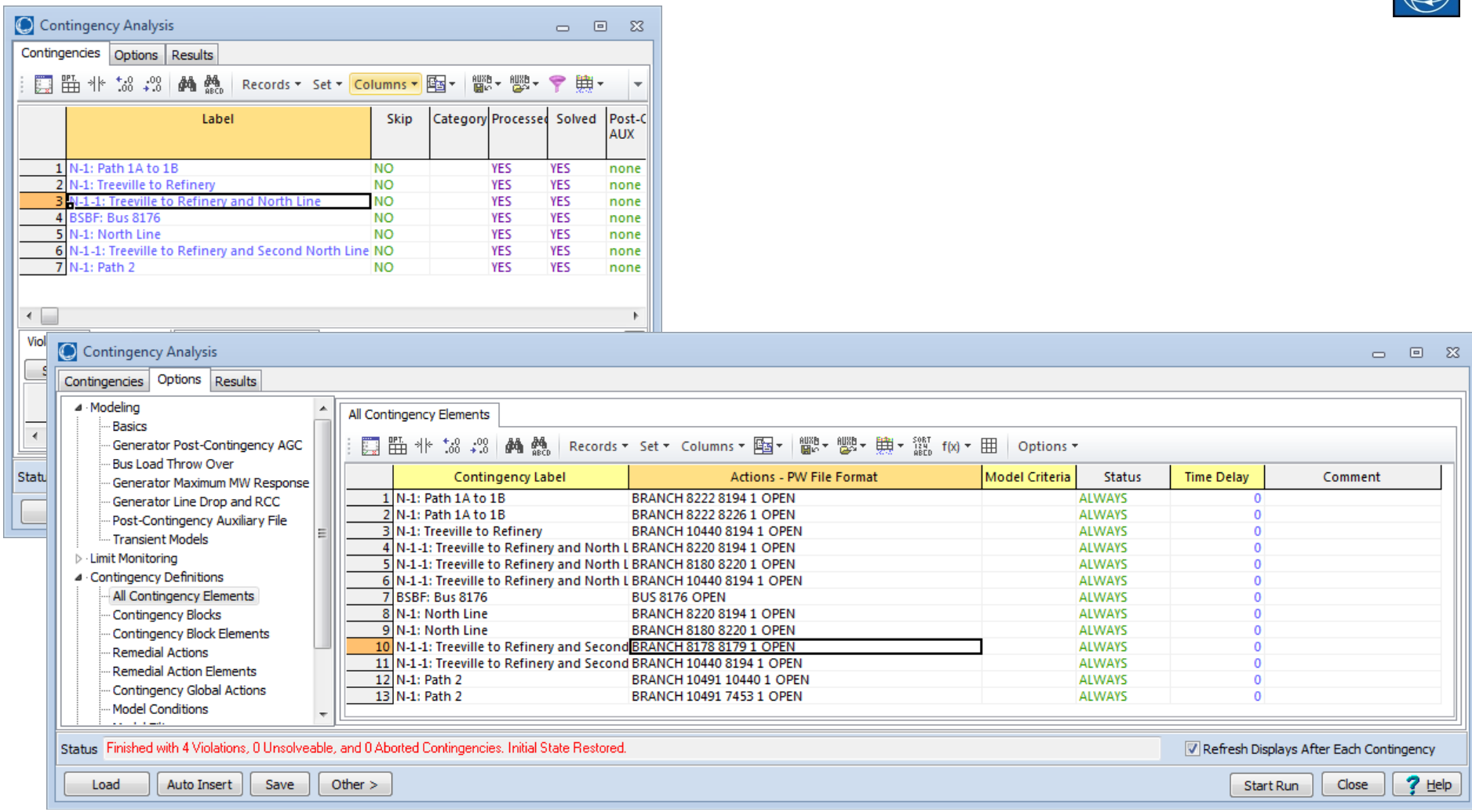
- *Object*
  - many choices of various contingency actions are available
- *Action*
  - What happens to the object
- *Criteria, CriteriaStatus*
  - Logical criteria under which actions are applied
- Time Delay
  - Use for ordering



ContingencyElement Actions

- There are many – we continue adding them as users have a need
  - Opening/Closing of transmission lines and transformers
  - Loss or Recovery of a generator, load, or switched shunt
  - Movement of generation, load, injection group, or switched shunt MWs or Mvars.
  - Changing or Setting of generation, load, injection group, or switched shunt MWs or Mvars
  - Changing or Setting of generator or switched shunt voltage setpoint
  - Opening of all lines connected to a bus
  - Opening of all lines connected to a substation
  - Opening/Closing of all lines or transformers in an interface
  - Open/Close, Set/Change injection group values
    - Many special options with this
  - Bypass/Inservice, Set impedance of series capacitors
  - Changing or Setting of phase-shifter setpoint
  - Open/Close 3-winding transformer
  - Open/Close DC lines, Set/Change DC line setpoints or resistance

Contingency Dialog



Contingency Definitions

- A lot more detail to discuss
  - We will cover in great detail on May 27 – 29 in Salt Lake City at WECC offices
- Final output for sharing with others

```
Contingency (Name, Category, Skip, Memo)
{
"L-2_Roughrider-Raven 2&3" "Double" "NO" "My Memo A"
"L-2_Roughrider-Raven 1&2" "Double" "NO" "My Memo A"
"L_Falcon-PatriotC1"       "Single" "NO" "My Memo A"
"T_Falcon-TitanC1"         "Single" "NO" "My Memo A"
}

ContingencyElement (Contingency, Object, Action, Criteria, CriteriaStatus,
                    TimeDelay, Comment)
{
"L-2_Roughrider-Raven 2&3" "BRANCH 15 54 2" "OPEN" "" "CHECK" 0 ""
"L-2_Roughrider-Raven 2&3" "BRANCH 15 54 3" "OPEN" "" "CHECK" 0 ""
"L-2_Roughrider-Raven 1&2" "BRANCH 15 54 1" "OPEN" "" "CHECK" 0 ""
"L-2_Roughrider-Raven 1&2" "BRANCH 15 54 2" "OPEN" "" "CHECK" 0 ""
"L_Falcon-PatriotC1"       "BRANCH 10 13 1" "OPEN" "" "CHECK" 0 ""
"T_Falcon-TitanC1"         "BRANCH 10 39 1" "OPEN" "" "CHECK" 0 ""
}
```

Contingency Solution: It's more than just the actions

- "Post-Transient" Power Flow: options matter a lot
  - What happens when you change MW injection by 2000 MW (outage generators)
    - Load or Generator change Make-up Power
    - Specify input parameter with each generator
      - Participation Factor
      - Max MW response
      - Disable response
  - Generator Voltage Control for Post-Transient
    - Generator Line Drop Compensation
    - Regulate terminal bus only (ignore remote regulation)
  - Bus Load Throwover (model distribution switching)
  - Control Options
    - Switched Shunt and Transformer switching

- I am skipping these options today, but we'll cover in detail at the Workshop in May. These options are important though!

Limit Monitoring Options

- These matter a great deal too
- Setup with the model (case)
  - Provide options to specify various limits for Branches (A, B, C, D, E, etc.)
  - Provide options for specifying various limits for Buses (A, B, C, D)
  - Provide LimitSet for choosing which limits to use
- Advanced Limit Monitoring During Contingency
  - Monitor for a violation due to a *change* in the system (5% voltage drop for example)

- I am skipping these options today, but we'll cover in detail at the Workshop in May

Accept Questions

- Pause for questions from audience

What do you need to model RAS in the Power Flow

- The description of a RAS is really the same as a Contingency.
  - A list of actions that occur
    - Actions become more complex though.
      - Trip MWs from a group of generators equal to 50% of the flow on an interface (or use a 2D lookup table to determine what to trip)
  - The *Criteria* is vital here
    → these actions do *not always* occur
    - Must describe the Boolean logic of when these actions occur
  - The CriteriaStatus and TimeDelay
    →describe when and at what point in solution process to include

- Question
  - *How do you implement the Boolean checks of when to trigger the RAS and the ordering of actions?*
  - *How do you handle the lookup tables, expressions used in more complex actions?*

Traditional Modeling of RAS in Software Studies

- Often the more complex features are provided *manually* by you the power engineer
  - Boolean logic of when to apply
    - May know that taking a double-line outage will cause RAS to be applied
    - Thus if contingency is for double line outage just include RAS actions
  - Figure out the "RAS Arming" level from the base case
  - Solve the contingency → If a line is overloaded then open it and resolve

Other Common Shortcuts

- Run a very detailed study of RAS
  - Particular contingencies cause violations (line overloads, bus voltage violations)
  - Verify that your RAS fixes these violations
- Then, for the next 1, 2, … (10?) years assume the RAS always works to fix these problems
- Functionally this means
  - Run your list of 100s or 1,000s of contingencies
  - Manually wade through the 1,000s of violations that occur and just ignore what is handled by RAS (experience)

Another common treatment for RAS

- Power engineer writes custom code to automate all the processes from the previous 2 slides
  - Write custom code to implement RAS
  - Write custom code to remove particular violations from your output reporting

Problems with these approaches (1/2)

- Reproducibility
  - Can you replicate a manual process and get the same answer over and over?
- Validity of assumptions
  - The RAS was designed many years ago. Are you sure your RAS is still always fixing the problems for which it was designed?
- Narrow assumption of when RAS is implemented
  - You are limiting the application of RAS to specific contingencies
  - This prevents you from seeing a cascading outage caused by several RAS interacting with each other

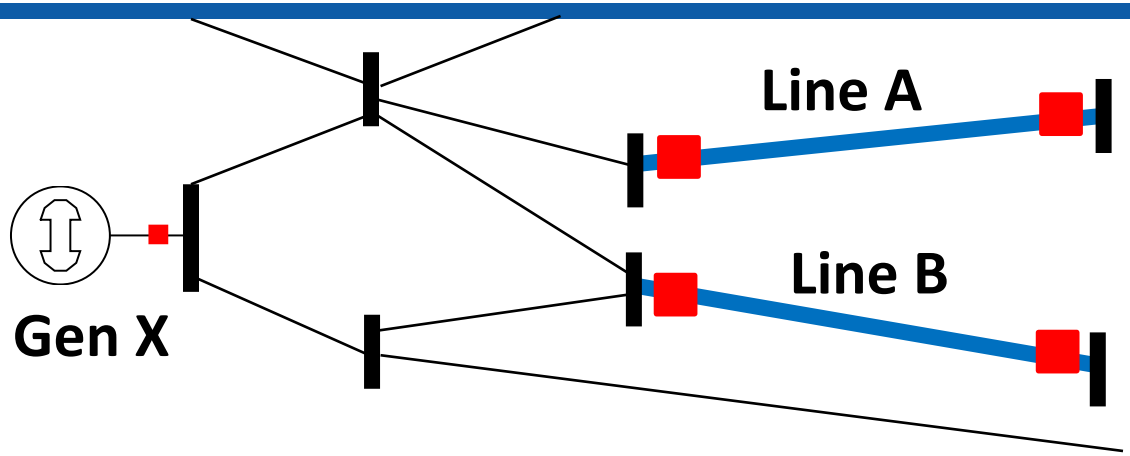
Problems with these approaches (2/2)

- Documentation and Sharing
  - How do you communicate what your RAS does to another utility or neighbor with the precision necessary to model it in software?
  - Custom code → who manages and takes support calls for that code?
- Input Data Management
  - Much of this leads to *manually* created contingency lists that are tuned for a particular operating condition
    - RAS arming, Boolean criteria
- Training – Human Resource Problem
  - Rely completely on the power engineer's experience which takes many years to develop
    - Engineers move jobs within a company
    - Engineers switch companies
  - How do you train new engineers or communicate all these assumptions? And do it quickly!

Another Hidden Problem

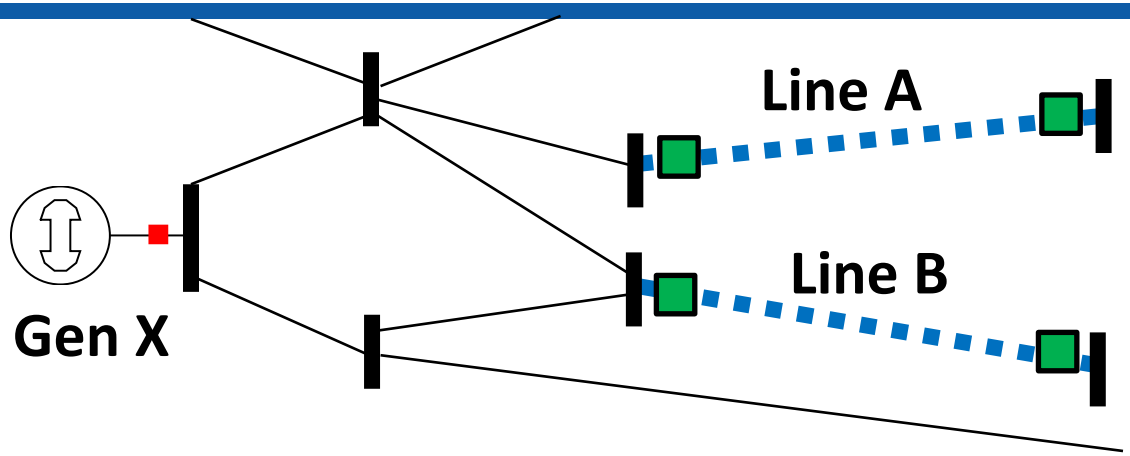
- PowerWorld's experience working with utility engineers
  - The engineers running power flow and transient stability studies have a general idea of how RAS functions
  - However, they may not manage and design the RAS itself, so they may miss details
- The implementation of *when* to "arm" and *how much* is very specific
  - The details matter!
  - Consider a very simple RAS example next

“Simple RAS”



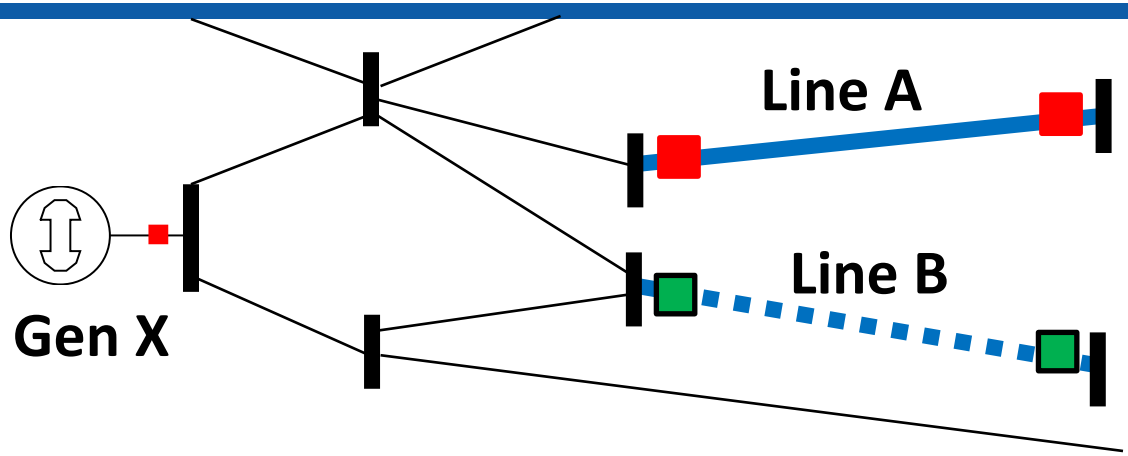
- General Description of RAS
  - If two transmission lines (**Line A** and **Line B**) are tripped → then trip a generator (**Gen X**)
- RemedialAction definition seems simple
  - *Object* = **Gen X**
  - *Action* = OPEN
  - *Criteria* = (**Line A** is OPEN) AND (**Line B** is OPEN)

Wait!
Differences in Initial Case



- What if **Line A** and **Line B** are out for maintenance this week?
  - Using our “Simple RAS” then this means that the Criteria always evaluates as TRUE!
  - Running a contingency analysis tool with this RAS defined would trip **Gen X** under every contingency
  - In this example, clearly the RAS criteria should evaluate to FALSE

Wait?
More differences in Initial Case



- What if **Line B** is out for maintenance this week?
- Using our "Simple RAS" then this means that if **Line A** is opened during the contingency then **Gen X** will be tripped
  - That *might* be correct, … but
  - It may *not* be correct for some RAS
    - For some if **Line B** is OPEN in the reference case then the RAS will *not be armed*! Tripping **Gen X** is not correct then
- The engineers running power flow/stability need to go talk with the RAS engineers to learn these details

Communication

- RAS design requires that their be redundant communication systems for RAS to prevent communication failure
- We also need the human communication between different groups of engineers to work as well
- Our experience is there are 3 groups here (though at smaller organizations there is overlap)
  1. “Planning or Operation Engineers” who run power flow and transient stability studies (these could be separate groups too)
  2. “Relay Engineers” who manage and design the system protection
  3. “RAS Engineers” who manage and design the RAS
- Most in audience today are in Group 1, but we all need to engage with folks in Group 2 and 3 to get the details of RAS and Relay modeling correct

Accept Questions

- Pause for questions from audience

Skip to Examples Today

- The next 30 slides give a broad overview of most of the various details that matter for RAS
  - This is the kind of material we will cover in detail during the Classroom Style portion of the Workshop in May
- For this presentation we are skipping this and will only briefly touch on the CriteriaStatus and using Stability Models in more detail
- I will then go through examples to demonstrate concepts

Skip to Slide #60

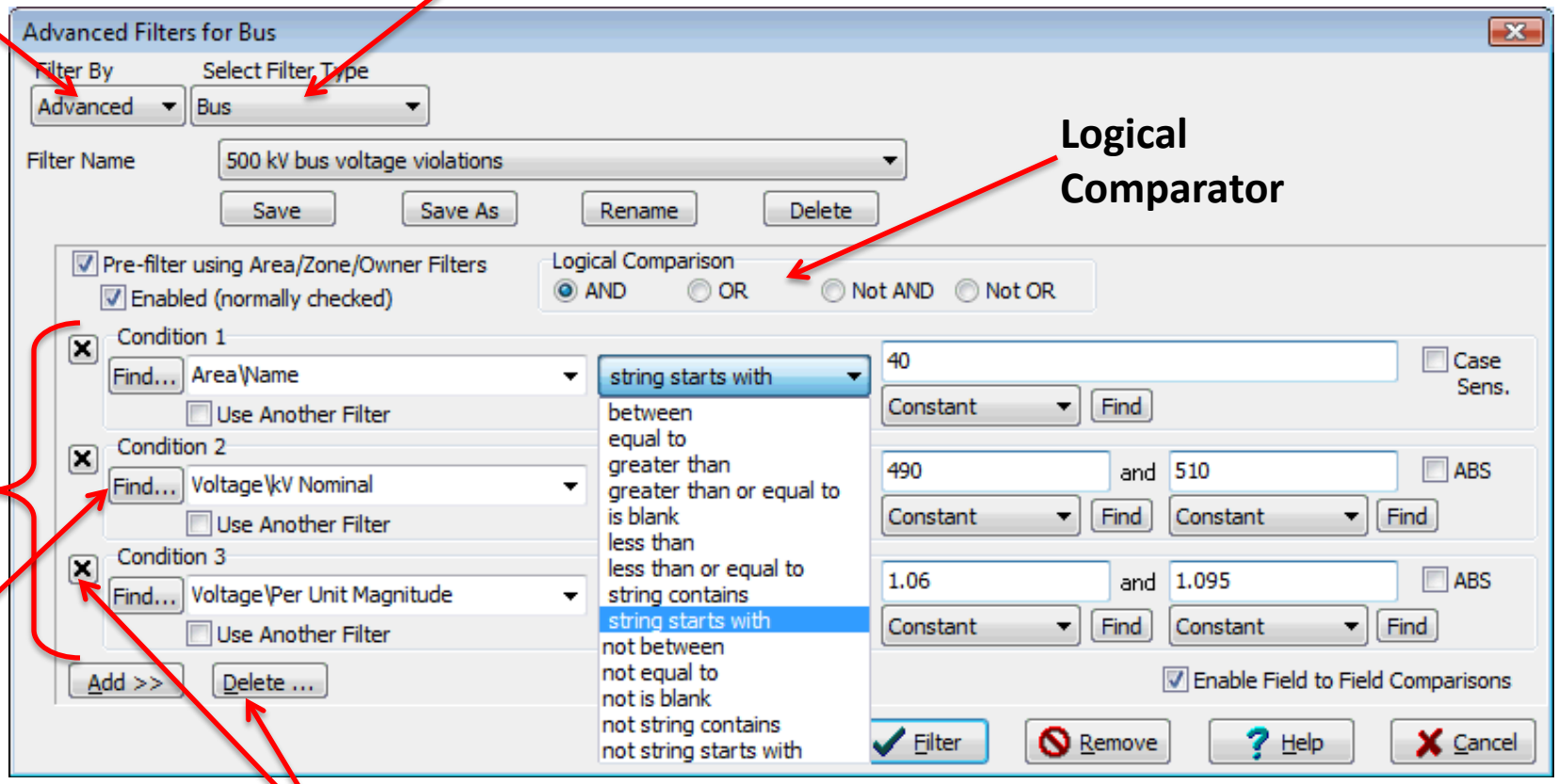
Specifics of Implementation

- Defining Criteria
  - Filtering
  - ModelCondition
  - ModelFilter
- Conditional Contingency Actions
- CriteriaStatus = *POSTCHECK*
- ModelExpression (lookup tables)
- RemedialAction
- Complexity of Injection Group Gen/Load dropping
- Injection Group Actions
- Overlapping Gen Drop
- Evaulate Model in Reference State
- CriteriaStatus = *TOPOLOGYCHECK*
- Disable if TRUE in Reference State
- Transient Stability Models in Power Flow Contingencies
- Time Delay
- Overall Contingency Process

Advanced Filtering
First Step for Conditional RAS

- Filtering is completely generic and available for all objects
  - Compare field to a constant (Flow > 500)
  - Compare field to another field (Flow > Limit)
  - Build any logical combinations of conditions and other filters
    - Be careful for circular references when using filters inside filters
  - No maximum on conditions
  - Compare across objects (Branches could be filtered based on whether the voltage was low at either end)

Advanced Filter Dialog



Filter By will be discussed shortly (Advanced or Device)

Type of object being filtered

Logical Comparator

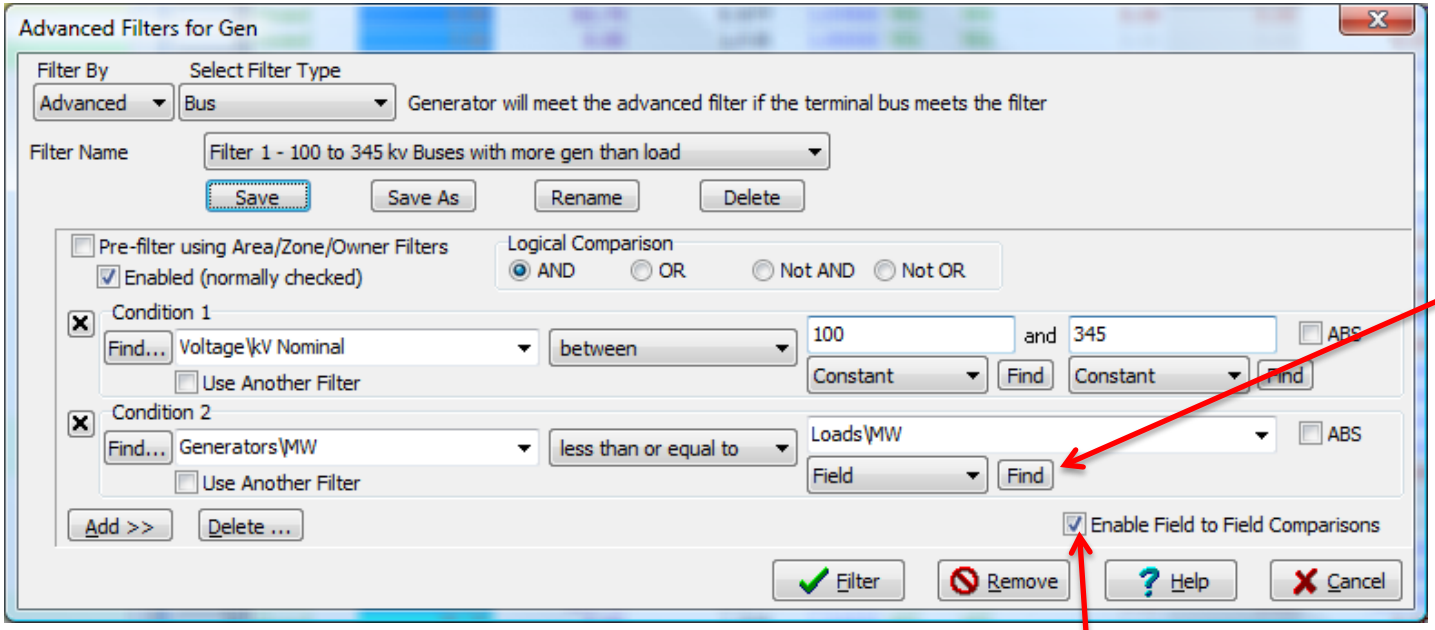
Conditions

Use the Find… buttons

Click to delete a condition

Field-To-Field Comparisons

- Users have the option to include Field-To-Field comparisons

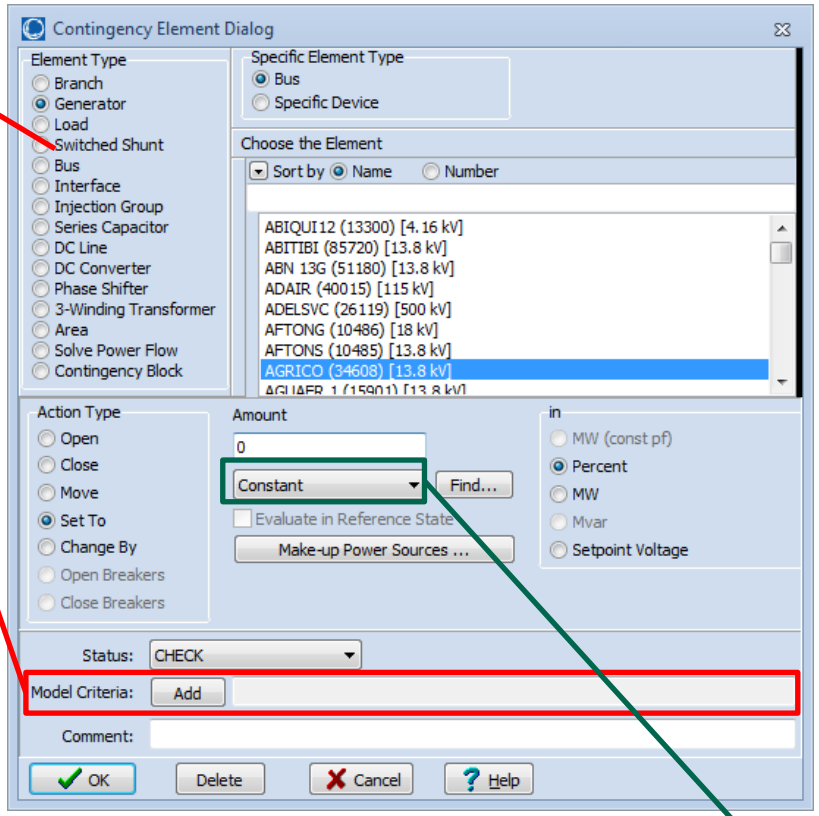


Click the Find Button to choose another field to compare to instead of a constant value.

Check the box for Enable Field to Field Comparisons

Conditional Contingency Actions

- Many choices of various contingency actions are available
- Model Criteria specifies a Boolean condition under which the action should be applied
  - Model Conditions
  - Model Filters
  - When a contingency is applied, these actions only occur if the Model Criteria is true



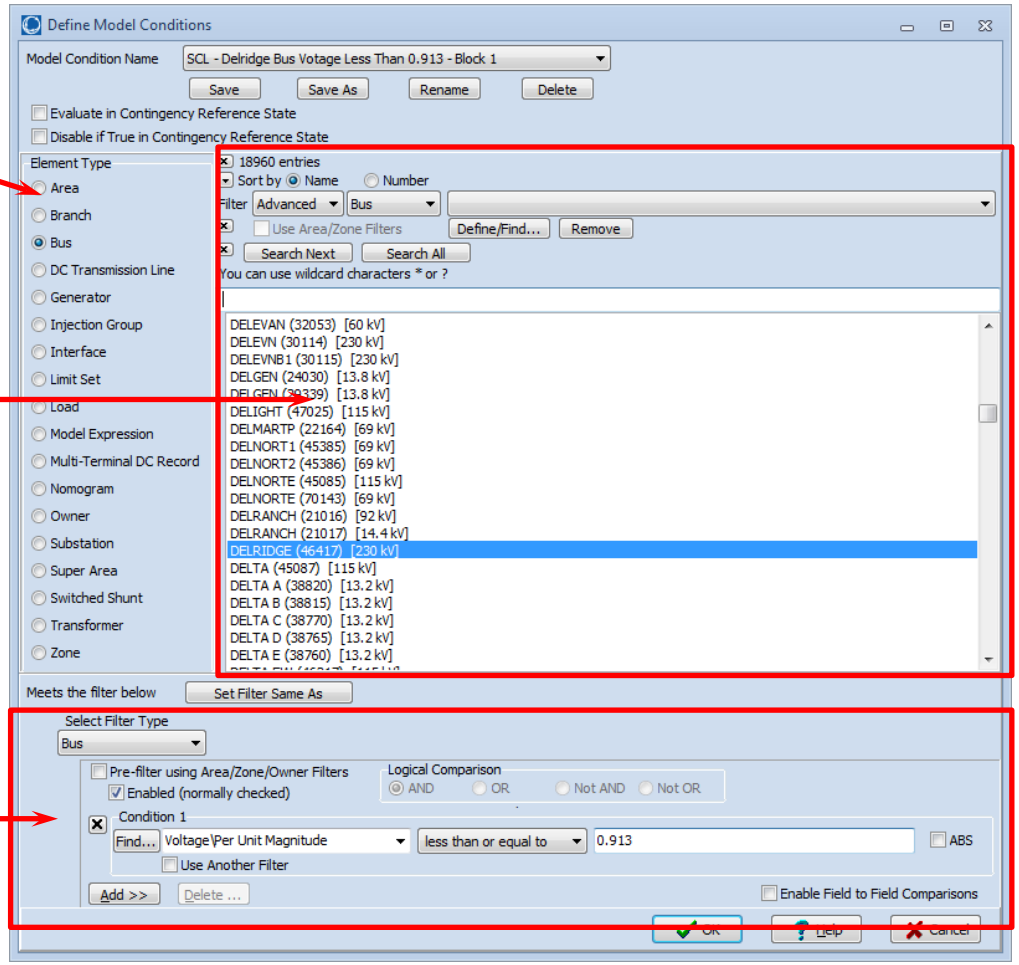
We'll discuss shortly

Model Conditions and Model Filters

- Model Condition (couples two things)
  - Particular object
  - An Advanced Filter definition
  - Model Condition is met if the advanced filter to the particular object is TRUE
  - TRUE Model Condition means action is applied
- Model Filter
  - A Boolean expression of other Model Conditions and Model Filters
  - TRUE Model Filter means action is applied

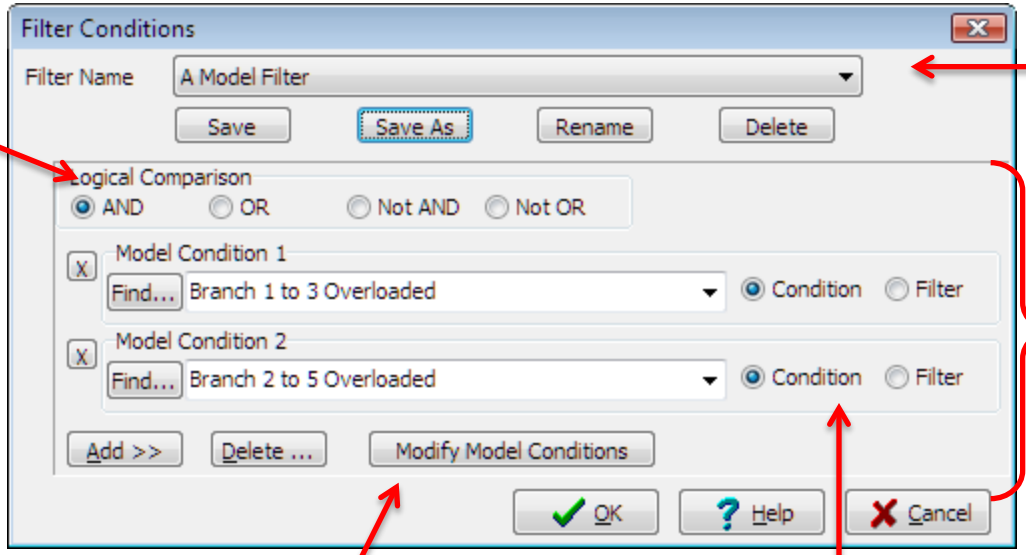
Conditional Contingency Actions
Model Conditions

- Choose Object Type
- Choose Object
- Build Advanced Filter Definition



Conditional Contingency Actions Model Filters

- Just piece together a list of Model Conditions or Filters and a logical comparison



Logical comparison for the Model Conditions

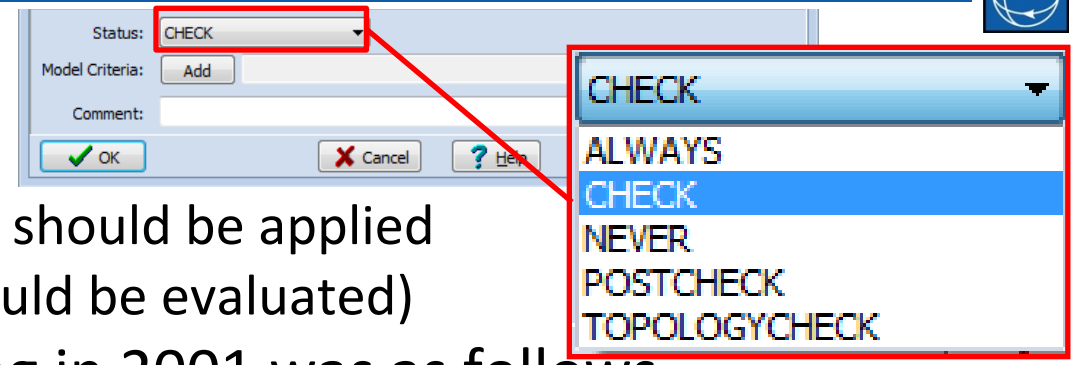
Give the Model Filter a name and save

Similar to the Advanced Filter Dialog

Click to Modify Model Conditions

Model Filters may contain Model Conditions or other Model Filters

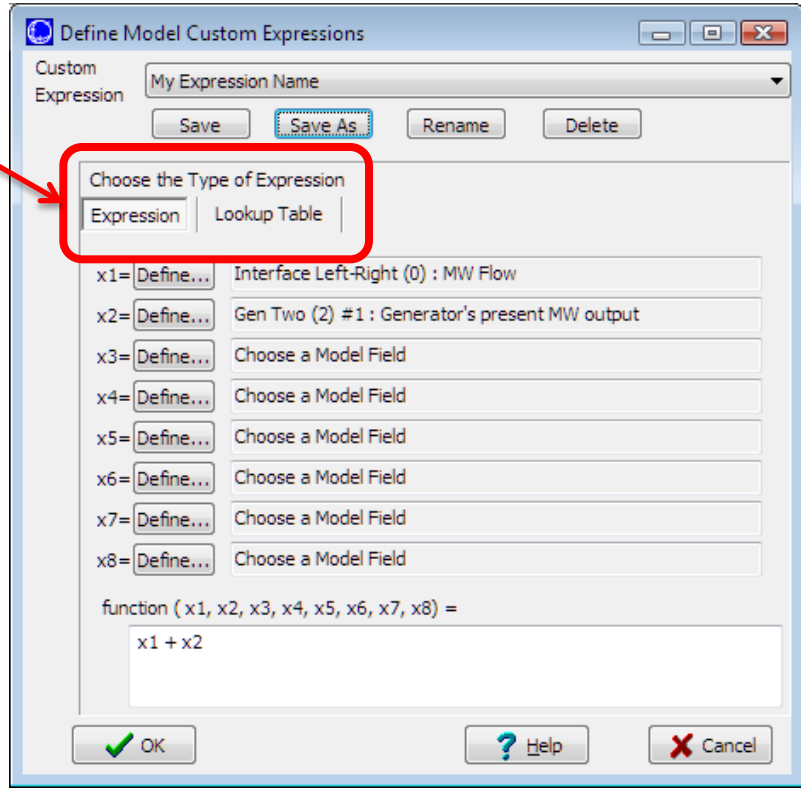
Conditional Contingency Actions: Action Status



- Action Status
  - Specifies when action should be applied
  - (i.e. when criteria should be evaluated)
- Contingency Processing in 2001 was as follows
  1. Apply *ALWAYS* actions and true *CHECK* actions
     - (Note: *CHECK* actions are evaluated in reference state)
  2. Update topology (branch, bus status)
  3. Solve power flow
  4. Apply true *POSTCHECK* actions
  5. Repeat steps 2-4 until no more *POSTCHECK* actions become true
- We will discuss *TOPOLOGYCHECK* shortly

Contingency Model Expressions

- Types
  - Expressions: mathematical expressions involving one or more model fields
    - Model Field can be any field of any object in the model
  - Lookup Tables: return a value based on the values of one or two model fields
- May be used inside
  - Some kinds of contingency elements
    - Set Gen MW to Model Expression
    - See earlier slide
  - Advanced Filters
  - Model Conditions
- Examples
  - Gen Drop Equal to a Lookup Table
  - Gen Drop Equal to 40% of Interface Flow

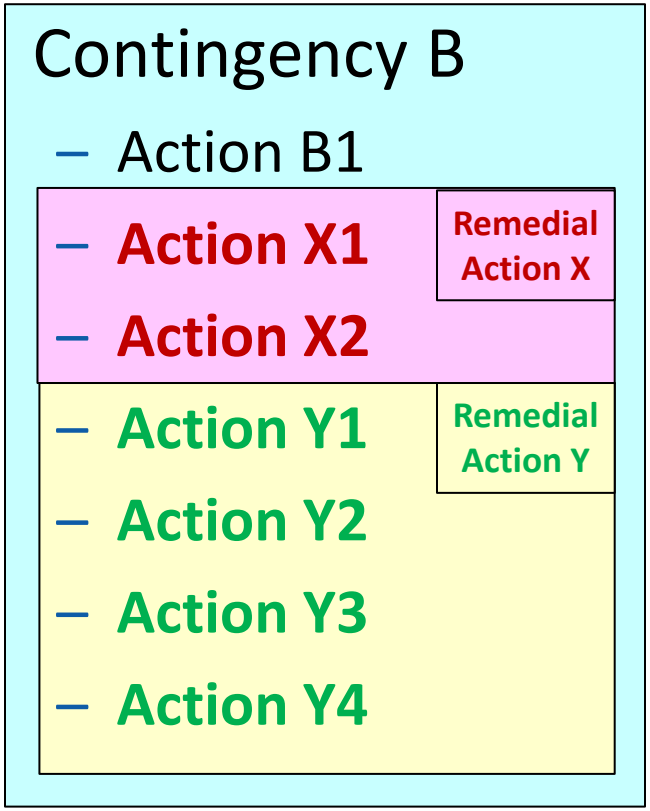


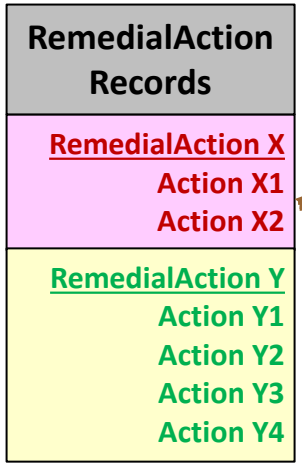
RemedialAction Objects

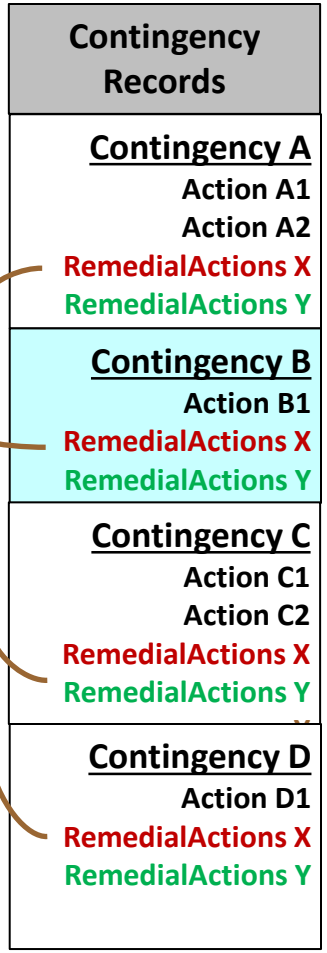
- The general logic shown still requires you to put all the actions inside of each contingency record
- **RemedialAction** records (and **RemedialActionElement** records)
  - This are a separate list of data record
  - They function the SAME AS a **Contingency** and **ContingencyElement** records
  - But every **RemedialActionElement** is automatically used as part of every **Contingency**

Contingency Records, **RemedialAction** Records

- End up with 7 actions actually used for a simulation of

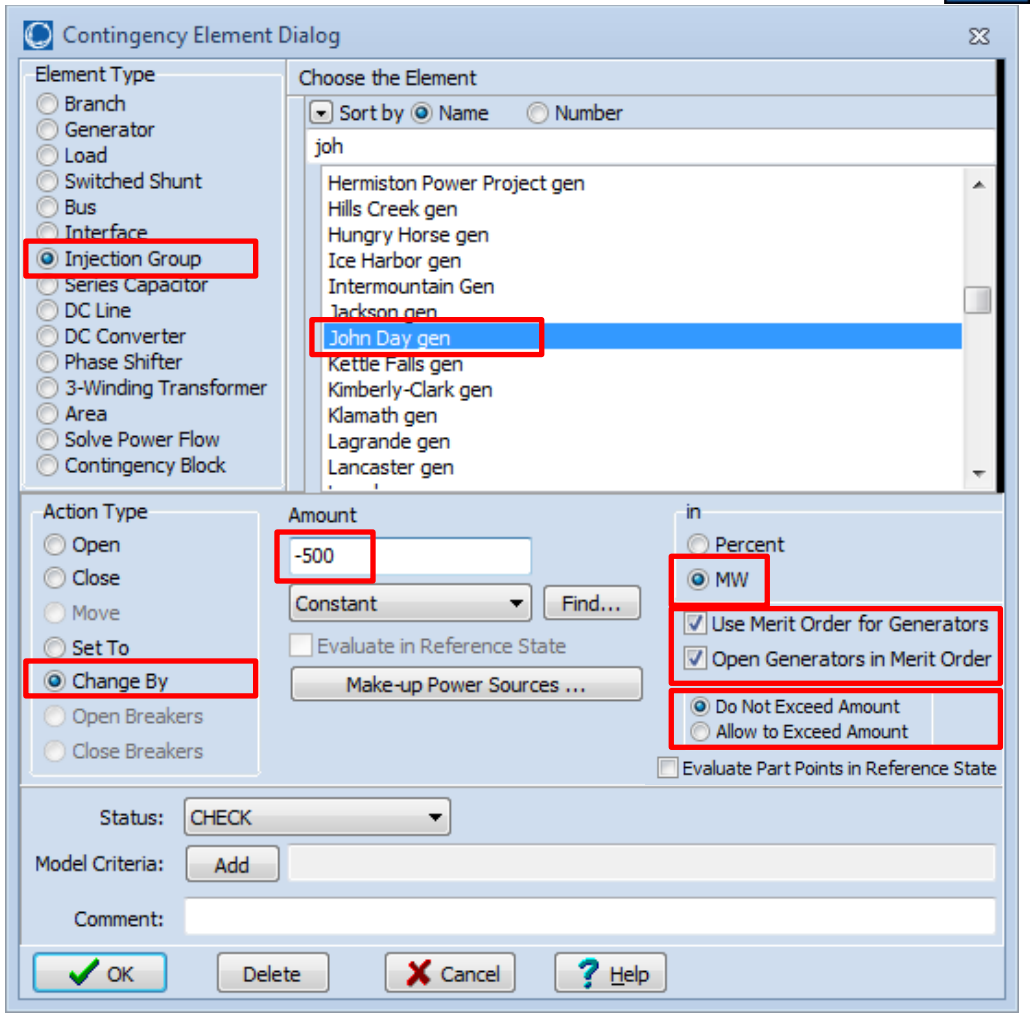






Example of Injection Group Contingency Action

- Assume Injection Group named John Day gen is defined
- Change the total John Day generation by -500 MW by opening generator in merit order



Contingency Analysis: Overlapping Gen Drop

- Accounting for Overlapping Generation when using merit-order generation dropping
  - Drop 500 MW from Injection Group #1
  - Drop 400 MW from Injection Group #2
  - Order matters: Group #1 will drop 500 MW, but when going to drop Group #2, if there is overlap and 200 MW of generation in Group #2 was already dropped due to Group #1 dropping, then only an additional 200 MW will be dropped.
  - You won't always get 900 MW of dropping (example above would only drop 700 MW)

- Note: default behavior is to take into account this overlap. This may be turned off in the Basic Contingency Modeling options

Evaluate Model in Reference State

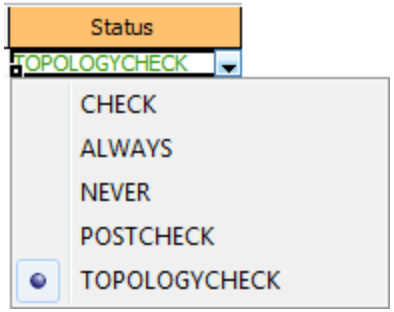
- Needed to arm the amount of generation drop based on *Reference State* only
  - Important if there are *POSTCHECK* actions
  - Possible that other *POSTCHECK* actions have occurred and you're iterating back on subsequent *POSTCHECK* actions with a new system state.
- Example:
  - Amount of generation drop is based on a two-dimensional nomogram which is a function of two interface flows
  - The "arming level" is handle based on the reference state, not what the MW flow happens to be at a particular point in contingency analysis process
    - Use Model Expression Lookup Table (Simulator supports 1D and 2D lookups tables)
    - For amount to drop, point to Model Expression
    - Check box for **Evaluate Model in Reference State**

Conditional Actions based on Status Only

- Users of Simulator had a persistent problem in some contingency runs when using RAS
  - RAS is configured to open 2,000 MW of generation when Line X trips
  - Before 2012, this was achieved by configuring a *POSTCHECK* action that looks at the branch status and trips generation if the branch is out of service
  - Works most of the time, but...
  - What if the outage of Line X results in an unsolvable steady state power flow solution?
    - Basically means that the RAS scheme is actually preventing a voltage collapse from occurring
    - This means that the *POSTCHECK* action is never evaluated because the power flow solution failed.
- Solution: *TOPOLOGYCHECK* actions

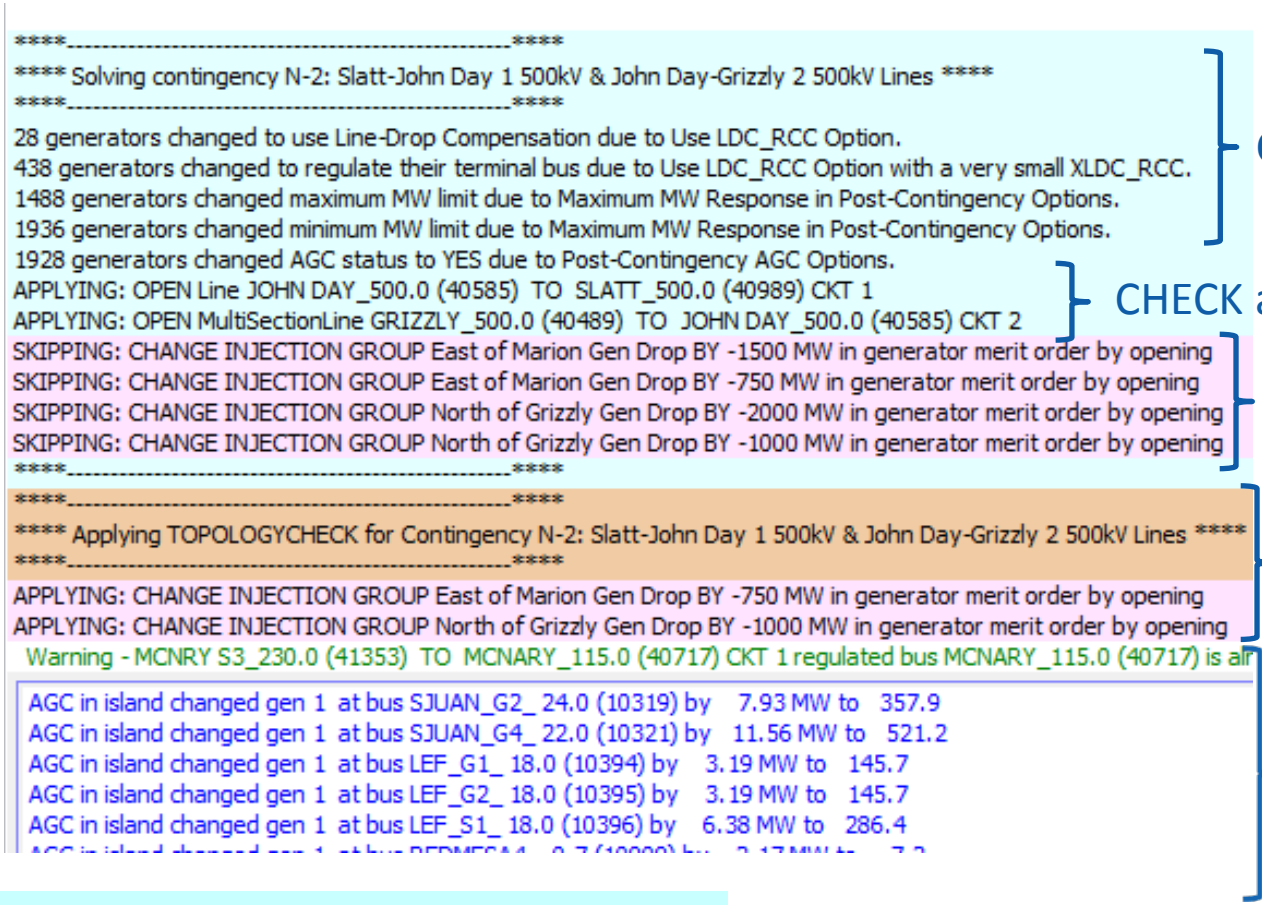
Contingency Analysis: TOPOLOGYCHECK

- ***TOPOLOGYCHECK*** Contingency Element Status.
- Contingency Processing now goes as follows
  1. Apply *ALWAYS* actions and true *CHECK* actions
  2. Update topology (branch, bus status)
  3. **ADD** Apply true *TOPOLOGYCHECK* actions
  4. Solve power flow
  5. Apply true *POSTCHECK* actions
  6. Repeat steps 2-5 until no more *POSTCHECK* and no *TOPOLOGYCHECK* actions are done



- Note: TOPOLOGYCHECK should only be used with Model Conditions related to bus/branch statuses

TOPOLOGYCHECK Status Message Log

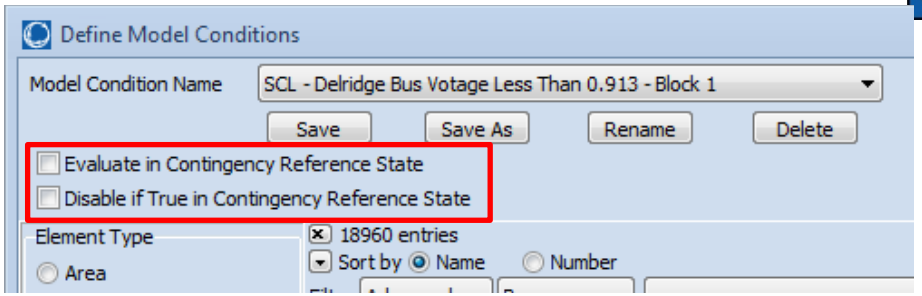


Light blue indicates contingency element

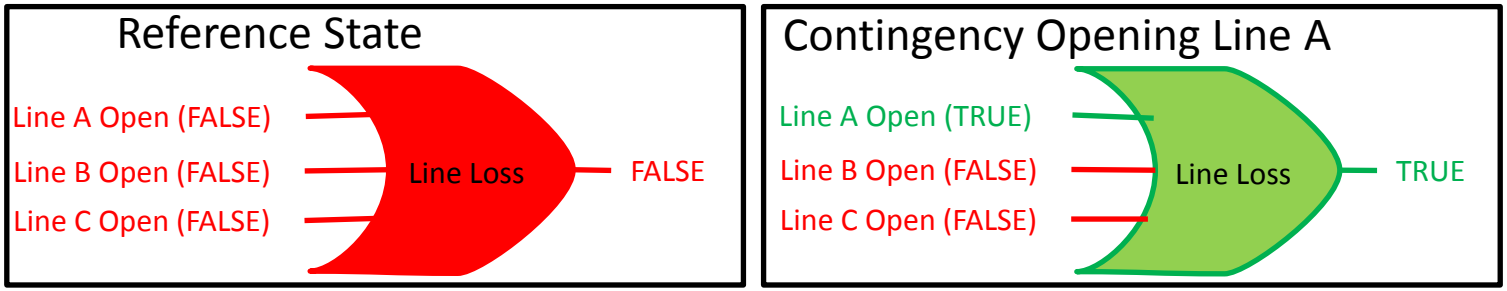
Pink indicates Global Action

Global RAS Modeling: Reference State Evaluations

- Model Condition Option
  - Evaluate in Contingency Reference State
  - Disable if True in Contingency Reference State



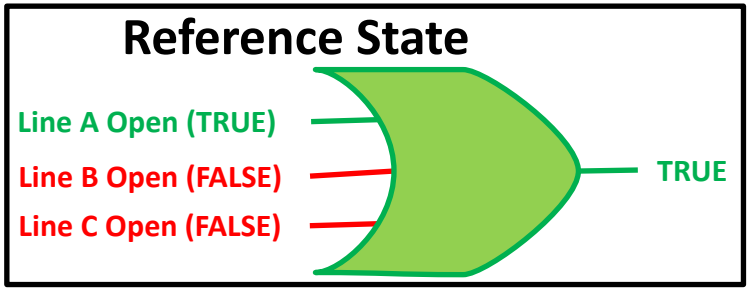
    - Example: Model Filter that takes the OR of three Model Conditions that test whether particular lines are Open
      - As soon as one of the lines is opened by a contingency, the model filter will evaluate to TRUE and you'll trigger appropriate actions.



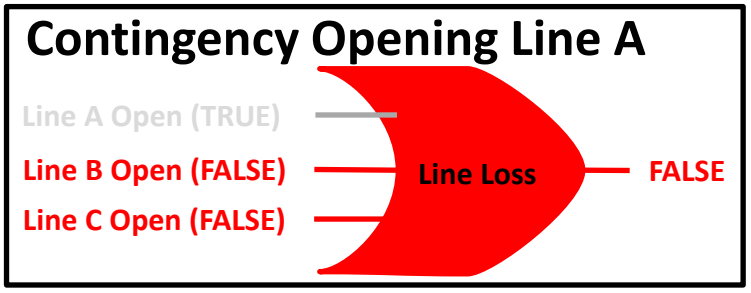
      - Works great as long as all three lines are CLOSED in the Reference State

Contingency Analysis: Disable if TRUE in Reference State

- What if Line A is OPEN in the Reference State?
  - This logic will end up returning TRUE for *EVERY* contingency



- Disable if TRUE in Reference State Option means
  - The Model Condition is completely ignored!
  - It's like the Model Condition doesn't even exist.
  - The result of this will depend on the type of logic the model condition is fed into



Contingency Analysis: Disable if TRUE in Reference State

- Reminder
  - Global Contingency Actions are a list of contingency elements that are included as part of every contingency solution
- Implication:
  - Specify an action as part of the Global Contingency Actions using the *Disable if TRUE in Reference State* choice
  - This allows you to model a global RAS in the power flow contingency action
    - Define one record in Global Contingency Action and you don't need to manage which contingencies use it
    - Also allows potential for cascading RAS under any contingency

Transient Relay Models in the Power Flow Contingency

- Conceptually a Transient Stability Relay Model in power flow contingency analysis act similarly to
  - Contingency action that opens a device (or devices)
  - **Status** = *POSTCHECK*
  - **Model Criteria =** Model Condition based on
    - Violating Bus Voltage Limit
    - Over Line Current
    - Apparent Impedance looking down line is inside Impedance Region (distance relay)
    - Voltage Limit for Load Relays as well
- Use in power flow contingency analysis
  - Force software to directly use *transient stability* relay models in *power flow* contingency analysis
  - Note: do NOT force extra definition of actions

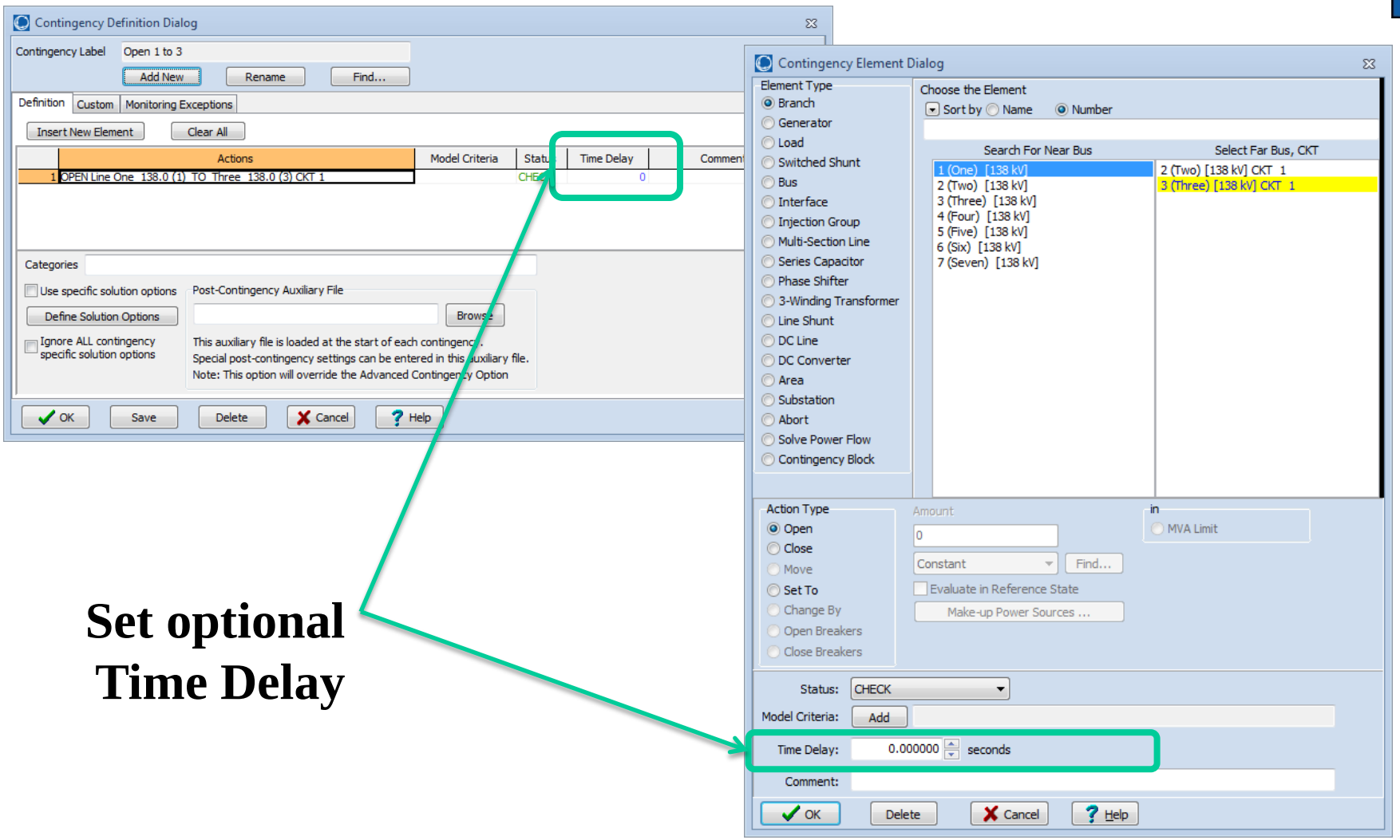
Transient Relay Models in the Power Flow Contingency

- Internally automatically evaluate steady-state implications of the stability relay models at the same time that existing *POSTCHECK* actions are evaluated
  - Assume in power flow contingency that post-contingency states exists forever
    - Timing data in relay models would be ignored as state exists forever
    - Over-current relays would just look at the minimum current threshold from transient model
    - Voltage based relays would look at largest minimum voltage and smallest maximum voltage
    - Distance/Impedance relays would evaluate highest zone for model
  - What Actually Occurred results would indicate if any of these actions are initiated
  - User Requirements for this feature
    - Define your stability relay models (Line and Load Relays)
    - Check a box to enable this new feature
- Auto-reporting options
  - Automatically report as a contingency violation if any relay models actual operate

Time Delay

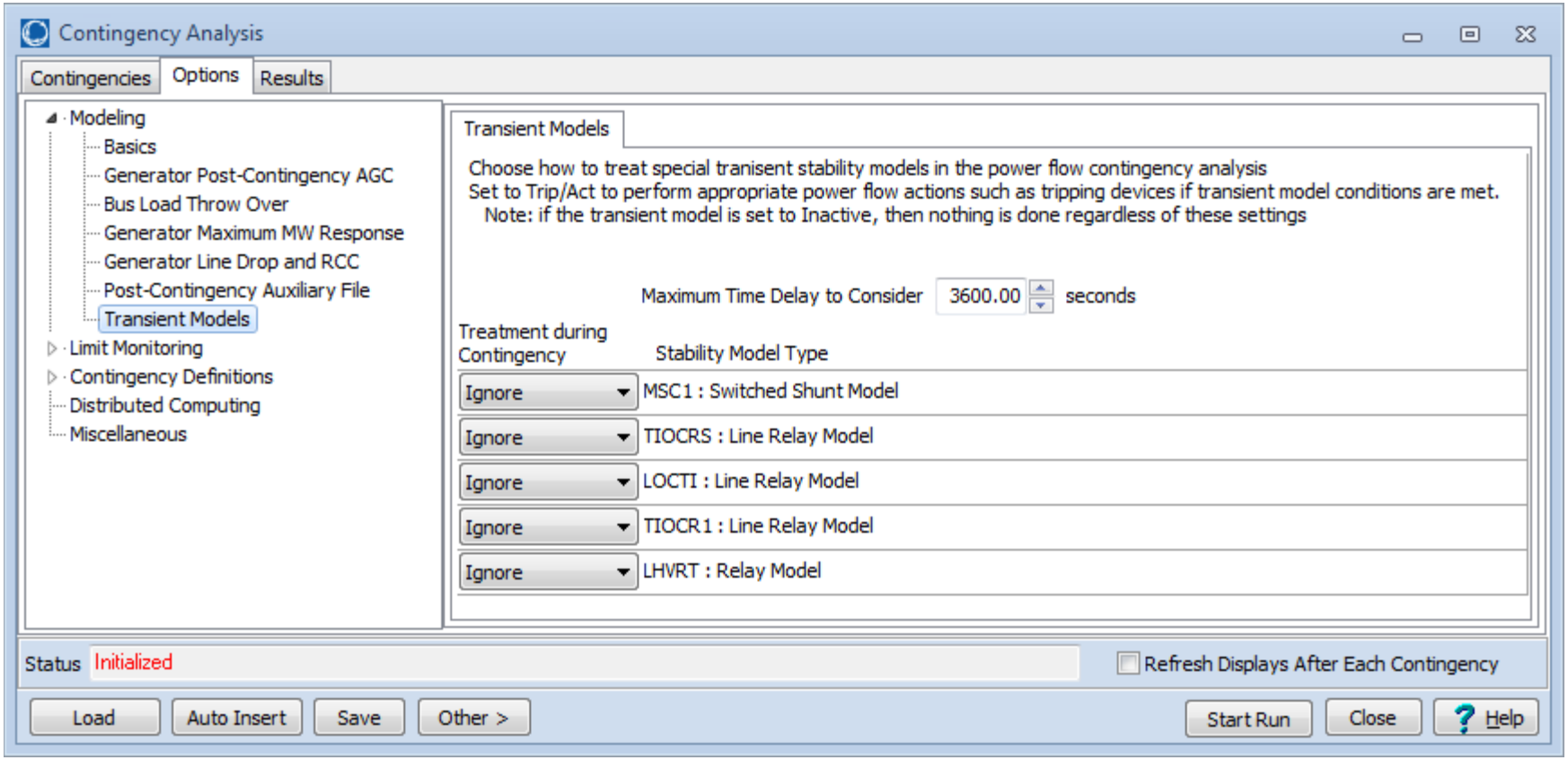
- Time to wait in seconds before an action is applied
- Serves as a relative ordering for implementation of actions during steady state analysis
- Actions with smallest delay (down to a microsecond) will be applied first during *TOPOLOGYCHECK* and *POSTCHECK* solution steps
- Ignored during *CHECK* solution step
- Default all time delays to 0 to ignore

Time Delay



Set optional Time Delay

Transient Models



Transient Models

- Several transient stability models can be included in steady state contingency analysis
- Options to specify how models are treated during steady state contingency analysis
  - Ignore
    - Nothing happens with this model during steady state contingency analysis
  - Trip/Act
    - If conditions are met, actual actions will be taken (such as tripping a line for overcurrent)
    - Some models have a Monitor flag. If this flag is set to monitor only, an individual model will not act regardless of contingency setting.
  - Monitor Only
    - If conditions are met, special contingency violations are reported
- Transient models are handled as part of the *TOPOLOGYCHECK* and *POSTCHECK* solution steps

Summary of Criteria Status and Transient Models

- Each **ContingencyElement** and **RemedialActionElement** have the following
  - Object (who to act upon)
  - Action (what to do)
  - Criteria (whether to apply)
  - TimeDelay (when to apply)
  - CriteriaStatus (where to apply in solution process)
    - *ALWAYS*, *NEVER*, *CHECK*
    - *TOPOLOGYCHECK*
    - *POSTCHECK*
- Transient Stability models can also be used in the power flow contingency processor
    - Call these *TRANSIENT* actions
    - Presentation today won't cover, but concept is the same to user

Processing Order of Contingency

1. Apply *ALWAYS* actions and true *CHECK* actions
2. Update topology (branch, bus status)
3. Apply true *TOPOLOGYCHECK* actions, and also any *TRANSIENT* actions
   - Only *TOPOLOGYCHECK* and *TRANSIENT* actions with smallest Time Delay
4. Solve power flow
5. Apply true *POSTCHECK* and true *TOPOLOGYCHECK* actions, and also any *TRANSIENT* actions
   - Only *POSTCHECK*, *TOPOLOGYCHECK*, and *TRANSIENT* actions with smallest Time Delay will be applied
6. If any *POSTCHECK*, *TOPOLOGYCHECK*, or *TRANSIENT* actions are implemented then repeat steps 2-6

- Keep repeating this over and over until no more actions occur

Accept Questions

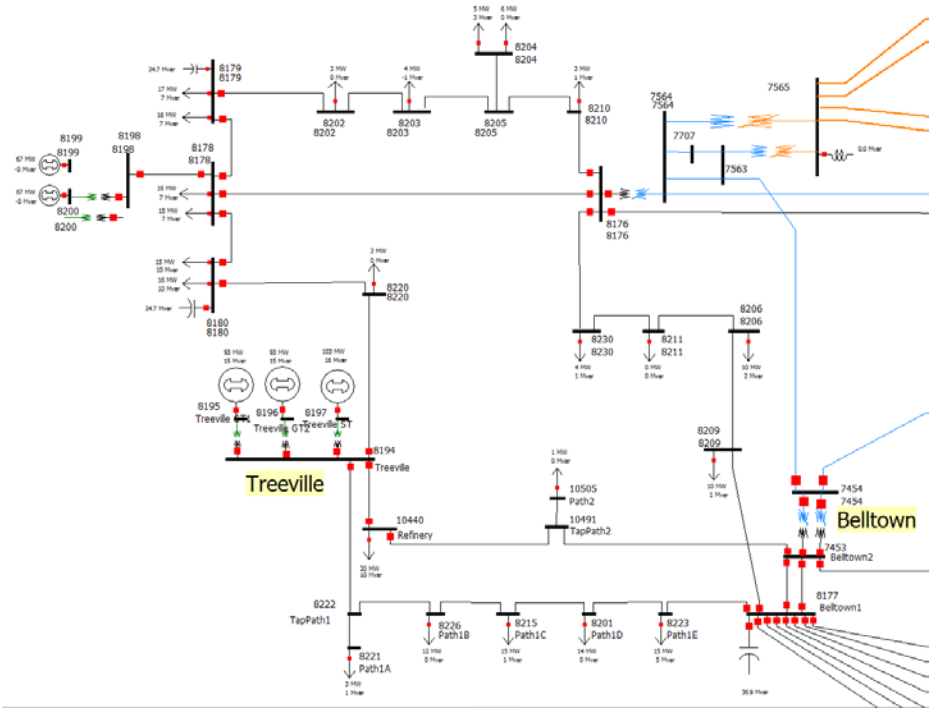
- Pause for questions from audience

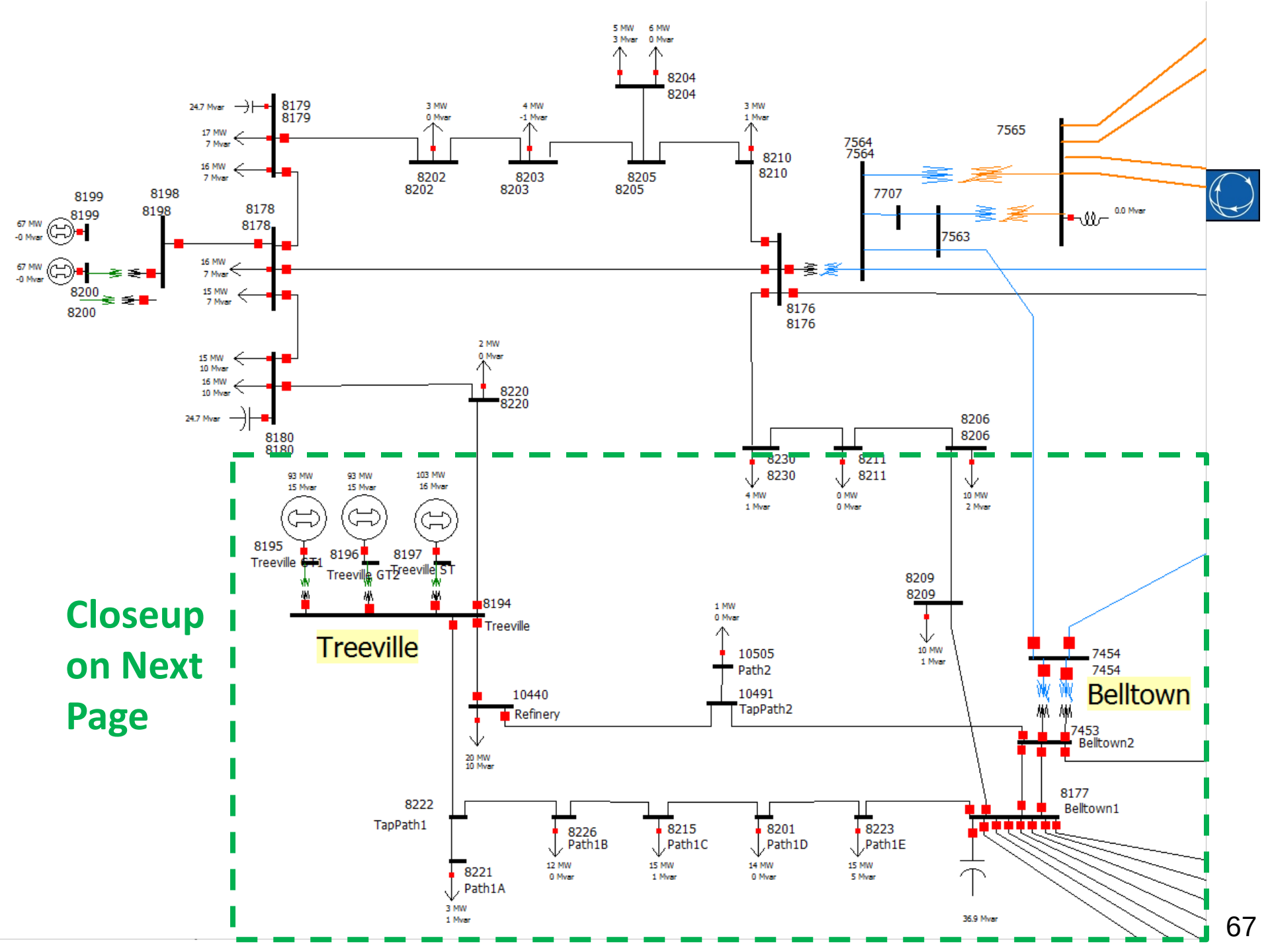
Two Example Remedial Action Schemes

- Complex Thermal RAS – “Treeville” RAS
  - This is a real RAS, but we’ve made up names so we can put this presentation on public websites
  - Complex Thermal-based RAS
  - Ordering of actions using Time Delay matters
- One of Two Line Outage with Generation Dropping
  - Handling the base case condition matters
  - Injection Group Dropping
  - Lookup Table
  - Merit Order
  - TOPOLOGYCHECK important

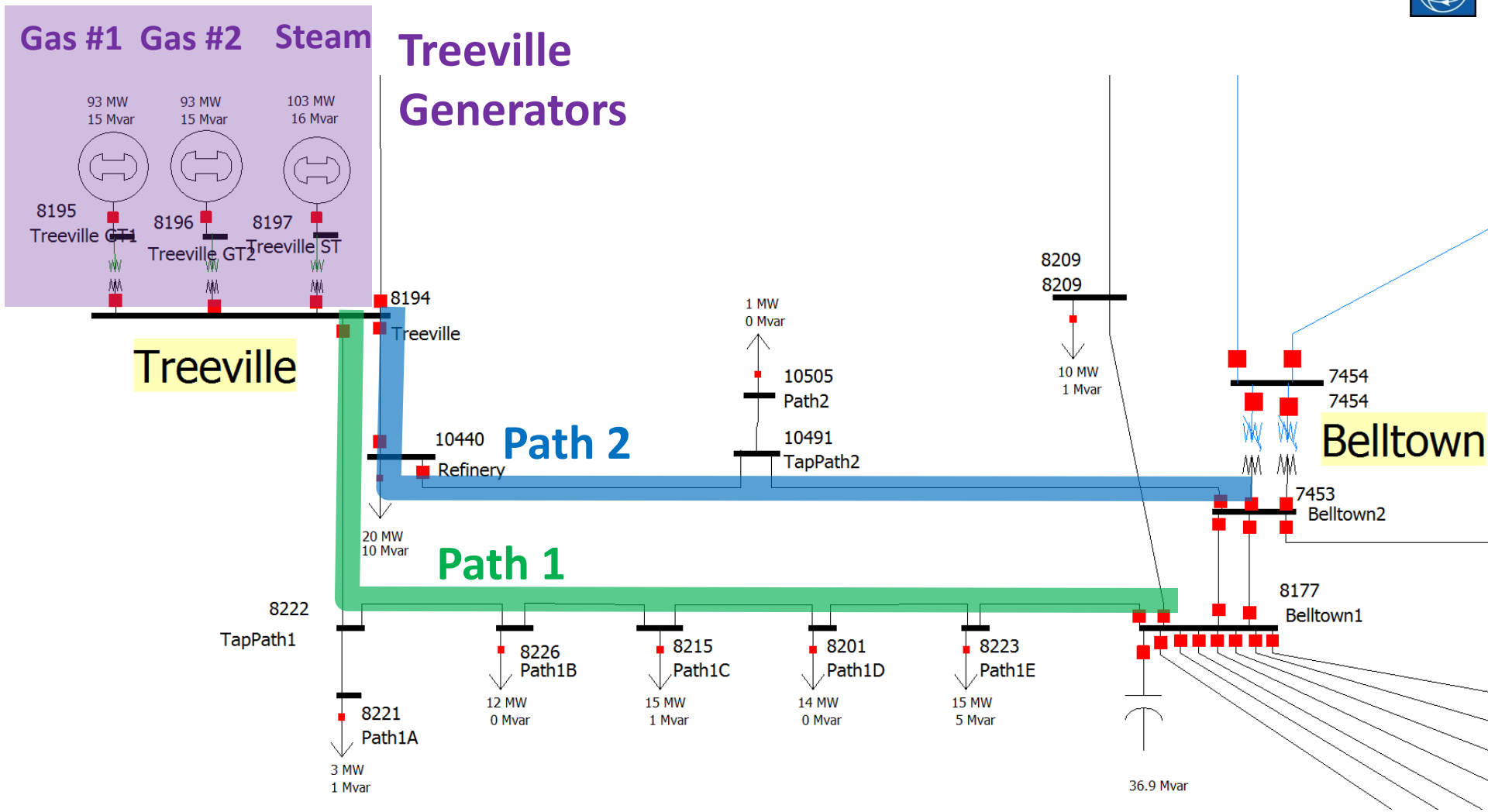
Treeville RAS

- Treeville RAS is used to relieve thermal overloads on 115 kV system around Treeville Generation
  - 2 Gas units (#1 and #2)
  - 1 Steam unit
- Two parallel paths between Treeville and Belltown




Closeup on Next Page
Treeville
Belltown
8204
8204
8202
8202
8203
8203
8205
8205
8210
8210
8179
8179
8199
8199
8198
8198
8178
8178
8200
8200
8176
8176
8220
8220
8180
8180
8230
8230
8211
8211
8206
8206
8195
Treeville GT1
8196
Treeville GT2
8197
Treeville ST
8194
Treeville
10440
Refinery
10505
Path2
10491
TapPath2
8209
8209
7564
7564
7565
7707
7563
7454
7454
7453
Belltown2
8177
Belltown1
8222
TapPath1
8221
Path1A
8226
Path1B
8215
Path1C
8201
Path1D
8223
Path1E

Treeville RAS Close-up



Note on Presentation

- For presentation of a concept, I will do the following
  1. Describe the RAS logic
  2. Show how that would be defined in a Dialog
  3. Show how that is represented in the RAS and Contingency File Format AUX file (text file)
- You'll see that the AUX file represents a bunch of tables
  - Power engineers are accustomed to tables of data about buses, gens, lines, areas, zones, owners, etc…
  - This is just a bunch of tables describing the logic, lookups, etc…

Treeville RAS Overview

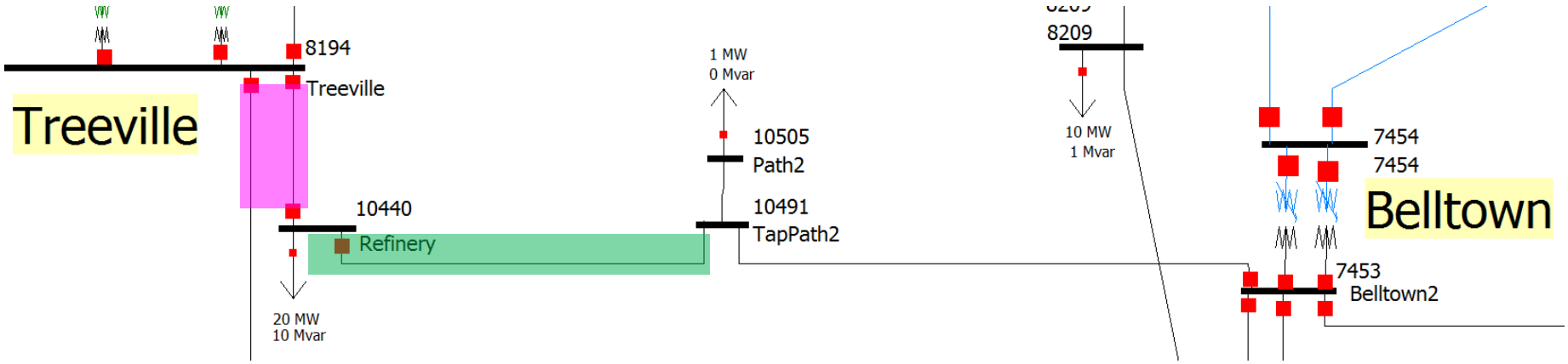
- 4 Separate Control Actions Possible
  1. Ramp Treeville generation down by 40 MW in two minutes
  2. Trip one Treeville gas combustion turbine
  3. Trip the all the Treeville generators
  4. Trip path of the 115 kV line going to the Refinery

- You will see Time Delay used for Actions 1-3
- Action 4 we'll cover first because it's so simple

Treeville RAS: Control Action #4

Refinery to TapPath2 > 110% of seasonal facility rating ⟶ Trip the Treeville to Refinery Line

- If Green Line Overloads then Trip Pink Line

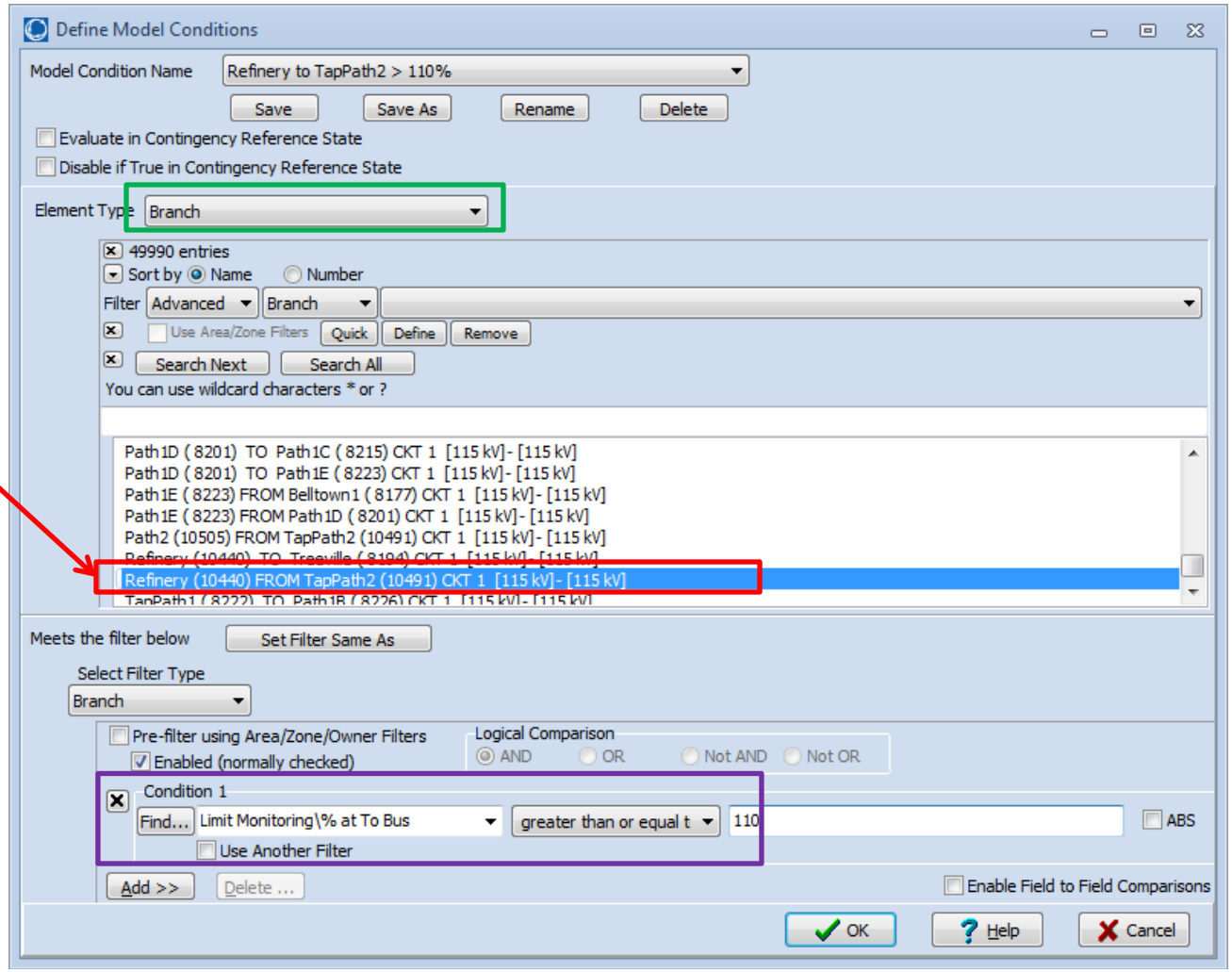


Treeville RAS Control Action #4: ModelCondition Dialog

Type = Branch

Choose Branch

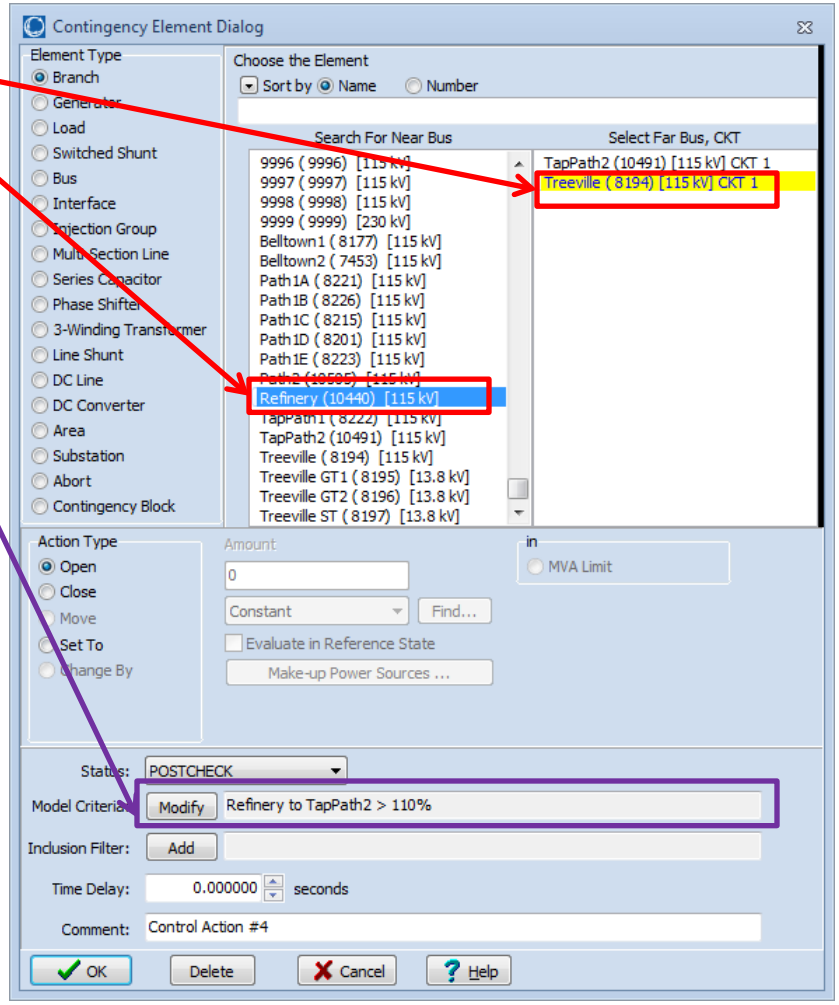
Condition for Branch



Treeville RAS Control Action #4: RemedialAction

Choose Branch to Open

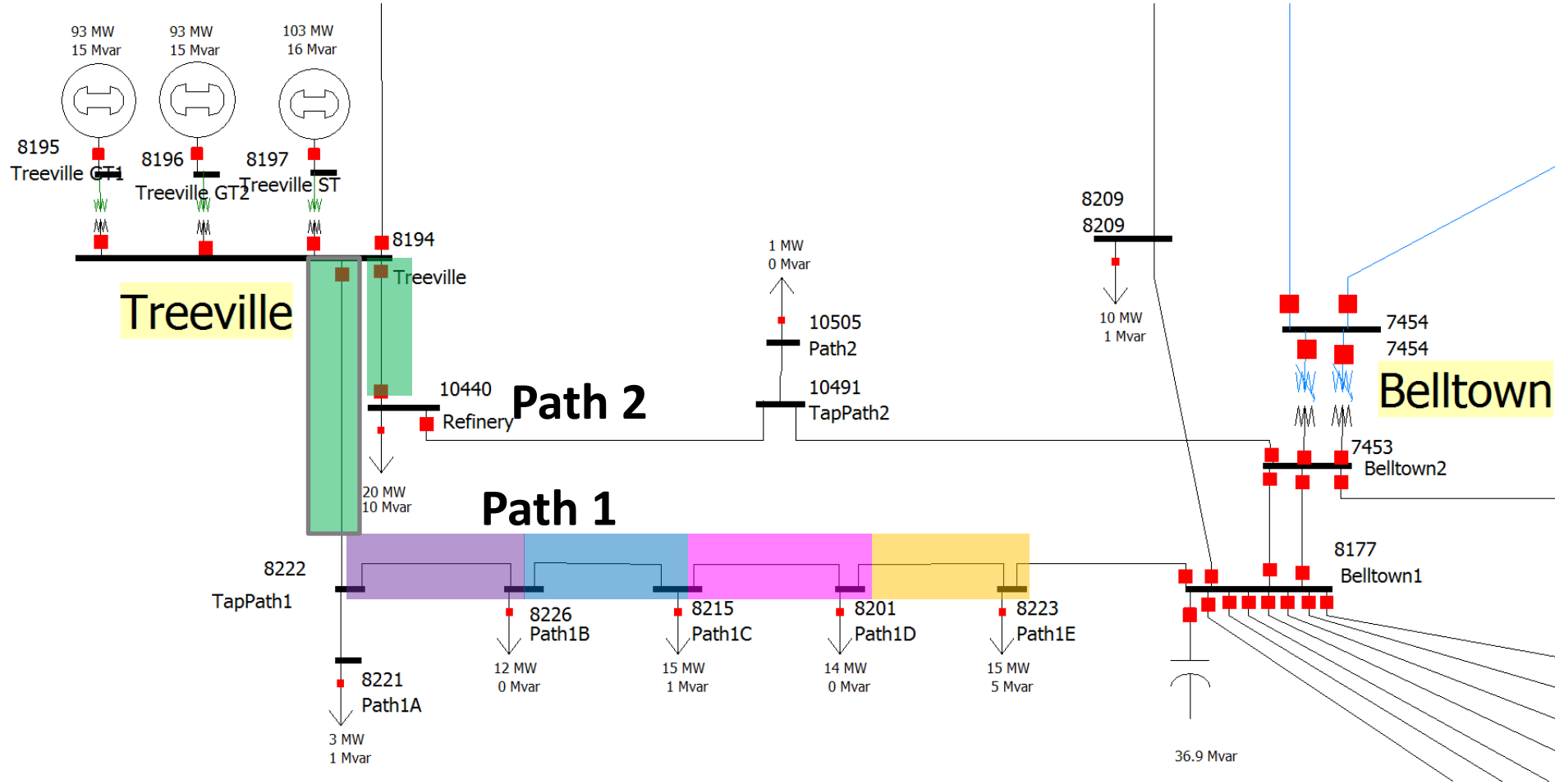
Model Criteria points to ModelCondition



Treeville RAS Control Action #4: File Format

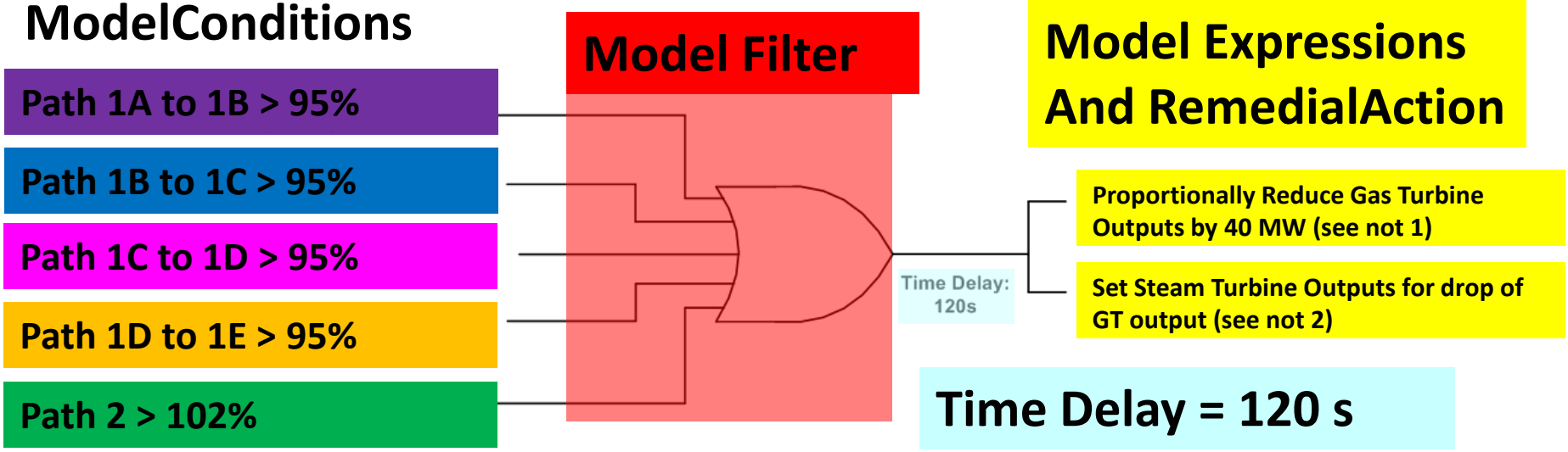
```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,
                EvaluateInRef,DisableIfTrueInRef,Memo)
{
"Refinery Tap2 > 110%" "Branch '10491' '10440' '1'" "Branch" "AND" "NO" "NO " ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,ObjectField,
                         ConditionType,Value,OtherValue,Absolute)
{
"Refinery Tap2 > 110%" 1 "Percent" ">=" "110" "" "NO "
}

REMEDIALACTION (Name,Skip,Memo)
{
"Treeville Generation Run-Back Scheme" "NO " ""
}
REMEDIALACTIONELEMENT (RemedialAction,Object,Action,Criteria,CriteriaStatus,
                       TimeDelay,InclusionFilter,Comment)
{
"Treeville Generation Run-Back Scheme" "BRANCH 10440 8194 1" "OPEN"
           "Refinery Tap2 > 110%"  "POSTCHECK" 0 "" "Control Action #4"
}
```

Color Codes for Treeville RAS Control Actions #1, #2, #3



Treeville RAS: Control Action #1

- Logic provided by utility that manages RAS



Notes:

1. CT model expressions are -(x1/(x1+x2)*40) for unit 1 and -(x2/(x1+x2)*40) for unit 2 where x1=actual MW output of unit 1 and x2=actual MW output of unit 2.

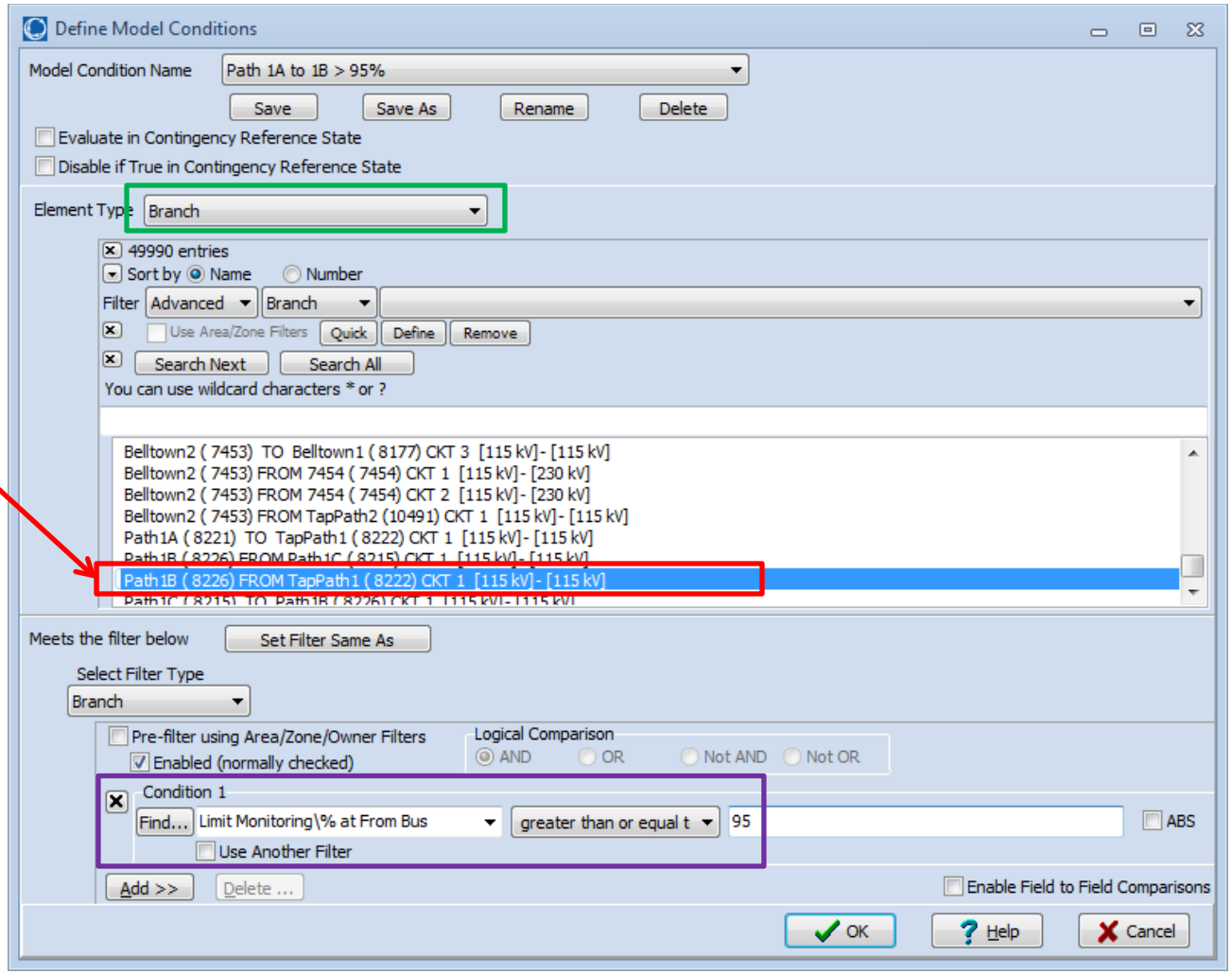
2. ST model expression is x5*((x1+x2-40)/(x3+x4)) where x1=actual MW output of unit 1, x2=actual MW output of unit 2, x3=Pmax of unit 1, x4=Pmax of unit 2 and x5=Pmax of steam unit.

Define Model Conditions: Path 1A to 1B > 95%

Type = Branch

Choose Branch

Condition for Branch

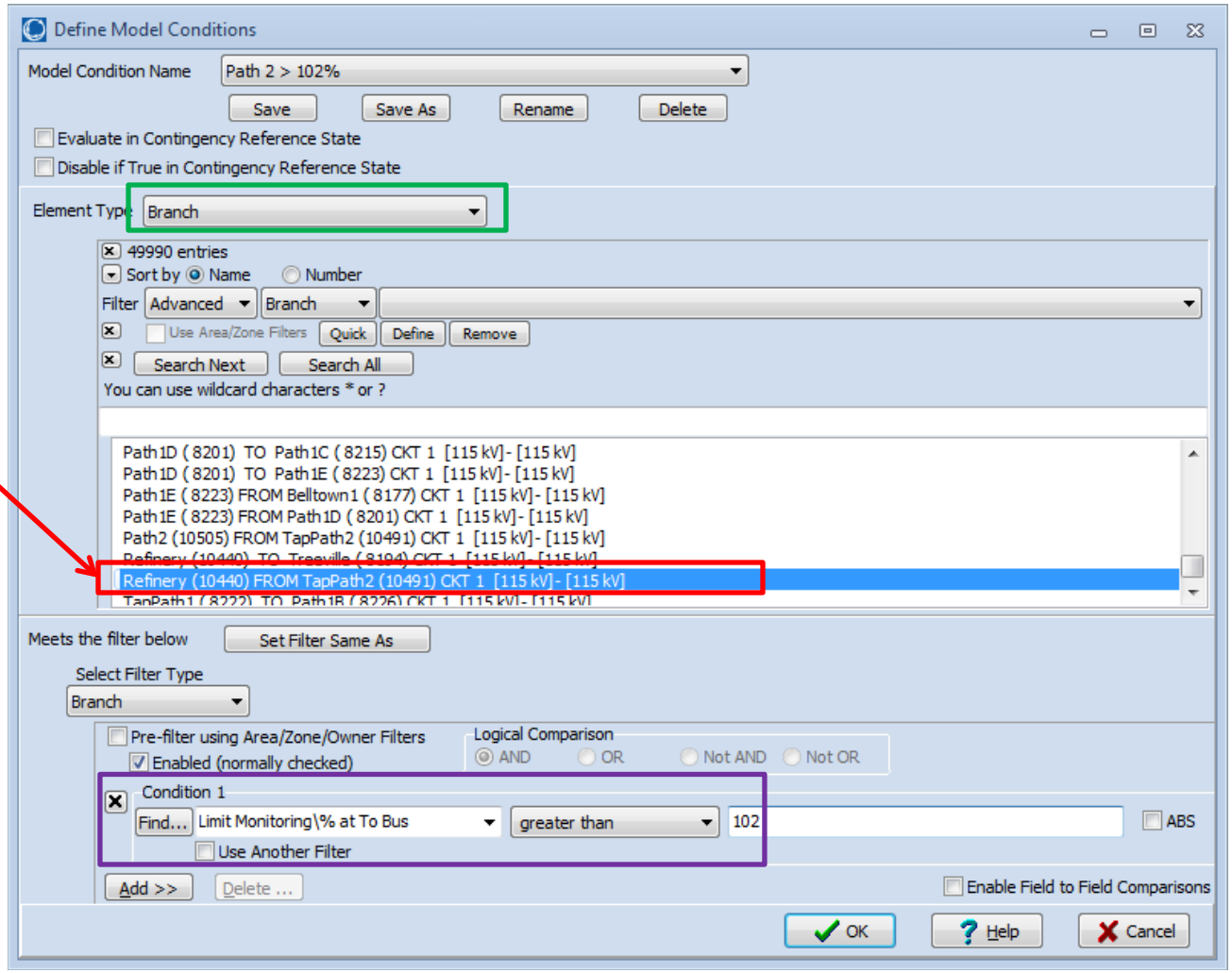


Define Model Conditions: Path 2 > 102%

Type = Branch

Choose Branch

Condition for Branch



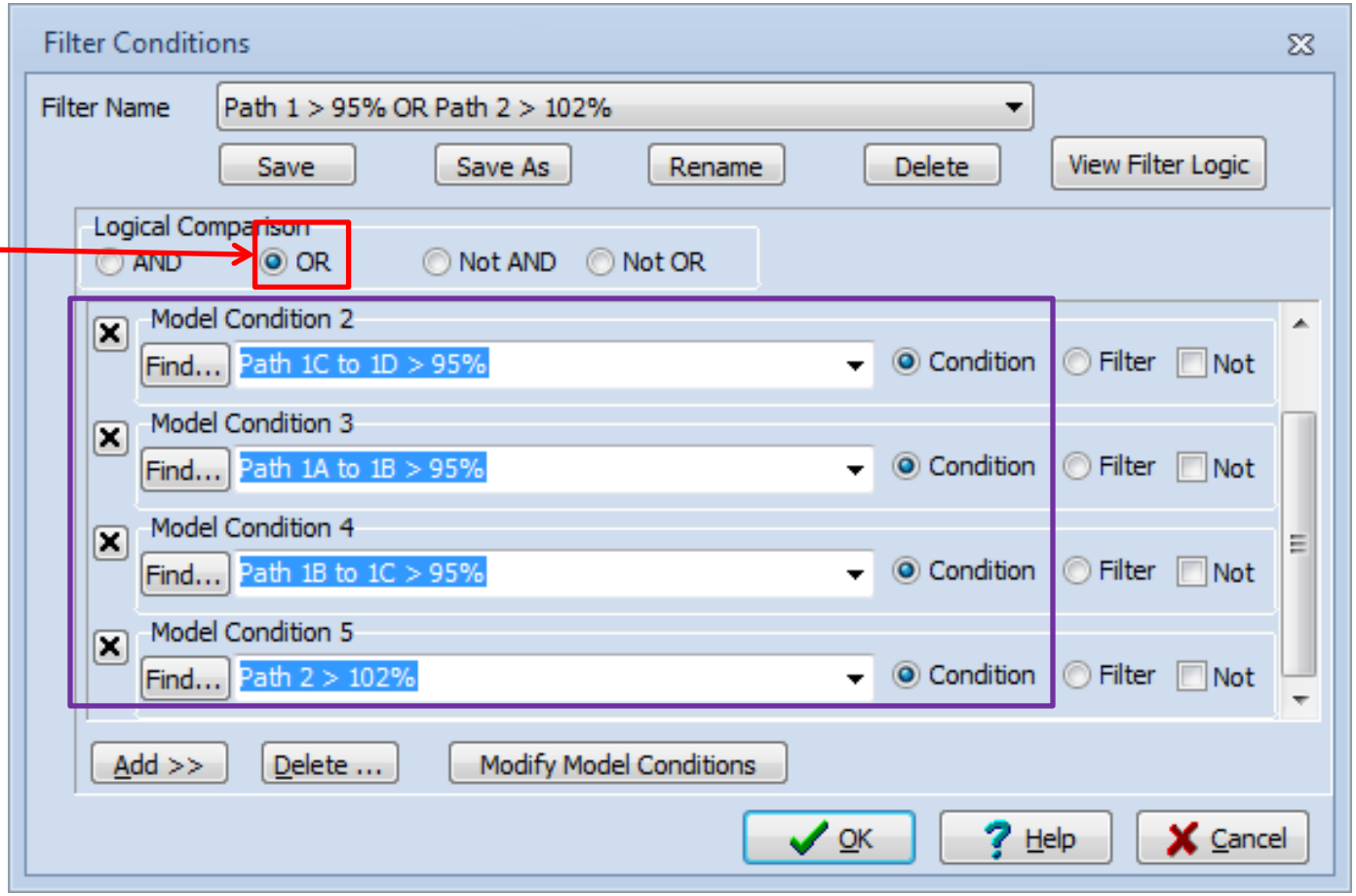
Treeville RAS Control Action #1: ModelCondition File Format

```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,EvaluateInRef,
                DisableIfTrueInRef,Memo)
{
"Path 1A to 1B > 95%"  "Branch '8222' '8226' '1'"   "Branch" "AND" "NO" "NO " ""
"Path 1B to 1C > 95%"  "Branch '8215' '8226' '1'"   "Branch" "AND" "NO" "NO " ""
"Path 1C to 1D > 95%"  "Branch '8201' '8215' '1'"   "Branch" "AND" "NO" "NO " ""
"Path 1D to 1E > 95%"  "Branch '8201' '8223' '1'"   "Branch" "AND" "NO" "NO " ""
"Path 2 > 102%"        "Branch '10491' '10440' '1'" "Branch" "AND" "NO" "NO " ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,ObjectField,ConditionType,Value,
                         OtherValue,Absolute)
{
"Path 1A to 1B > 95%"  1 "Percent" ">=" "95" "" "NO "
"Path 1B to 1C > 95%"  1 "Percent" ">=" "95" "" "NO "
"Path 1C to 1D > 95%"  1 "Percent" ">=" "95" "" "NO "
"Path 1D to 1E > 95%"  1 "Percent" ">=" "95" "" "NO "
"Path 2 > 102%"        1 "Percent" ">" "102" "" "NO "
}
```

Treeville RAS Control Action #1: ModelFilter Dialog

OR Logic

List of Model Conditions

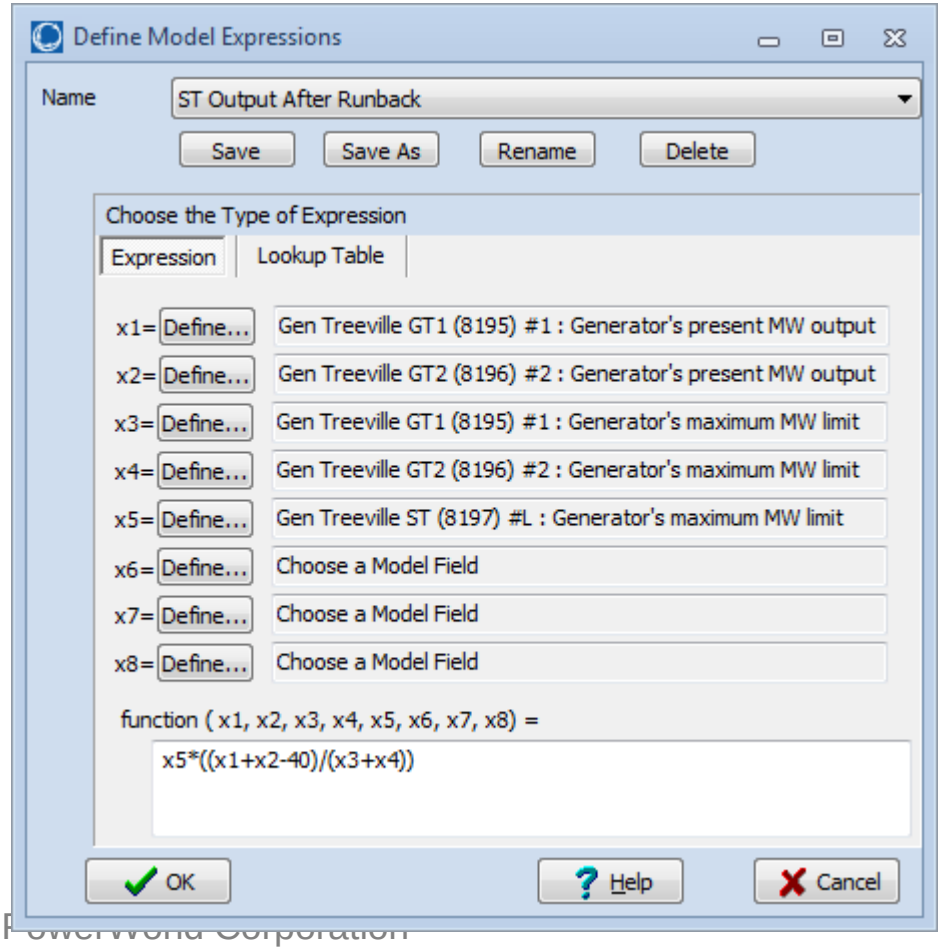


Treeville RAS Control Action #1: ModelFilter File Format

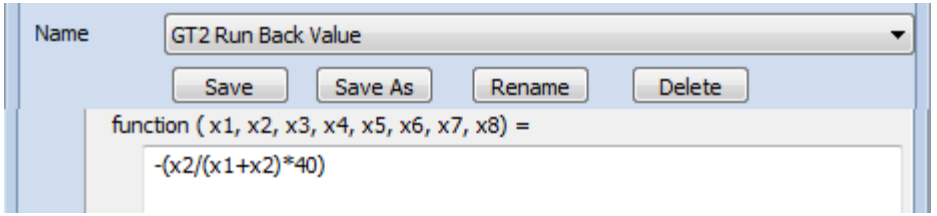
```
MODELFILTER (Name,Logic,Memo)
{
"Path 1 > 95% OR Path 2 > 102%" "OR" ""
}
MODELFILTERCONDITION (ModelFilter,CondNum,Criteria,Logic)
{
"Path 1 > 95% OR Path 2 > 102%" 1 "Path 1A to 1B > 95%" ""
"Path 1 > 95% OR Path 2 > 102%" 2 "Path 1B to 1C > 95%" ""
"Path 1 > 95% OR Path 2 > 102%" 3 "Path 1C to 1D > 95%" ""
"Path 1 > 95% OR Path 2 > 102%" 4 "Path 1D to 1E > 95%" ""
"Path 1 > 95% OR Path 2 > 102%" 5 "Path 2 > 102%"       ""
}
```

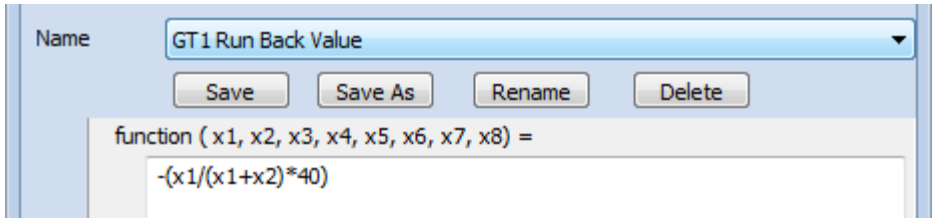
Treeville RAS Control Action #1: Model Expression

Move Steam Plant by the same net MW in proportion to Max MW



Reduce the net MW output of Gas Unit 1 and 2 by a 40 MW (move proportional to present output)





Treeville RAS Control Action #1: ModelExpression File Format

```
MODELEXPRESSION (Name,Type,Expression,Memo,
                 Object1,x1,BlankZero1,Object2,x2,BlankZero2,
                 Object3,x3,BlankZero3,Object4,x4,BlankZero4,
                 Object5,x5,BlankZero5,Object6,x6,BlankZero6,
                 Object7,x7,BlankZero7,Object8,x8,BlankZero8)
{
"GT1 Run Back Value"       "Expression" "-(x1/(x1+x2)*40)" ""
                "Gen '8195' '1'" "MW" "YES" "Gen '8196' '2'" "MW" "YES"
                "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO "

"GT2 Run Back Value"       "Expression" "-(x2/(x1+x2)*40)" ""
                "Gen '8195' '1'" "MW" "YES" "Gen '8196' '2'" "MW" "YES"
                "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO "

"ST Output After Runback"  "Expression" "x5*((x1+x2-40)/(x3+x4))" ""
                "Gen '8195' '1'" "MW" "YES" "Gen '8196' '2'" "MW" "YES"
                "Gen '8195' '1'" "MWMax" "YES" "Gen '8196' '2'" "MWMax" "YES"
                "Gen '8197' 'L'" "MWMax" "YES" "" "" "NO " "" "" "NO " "" "" "NO "
}
```

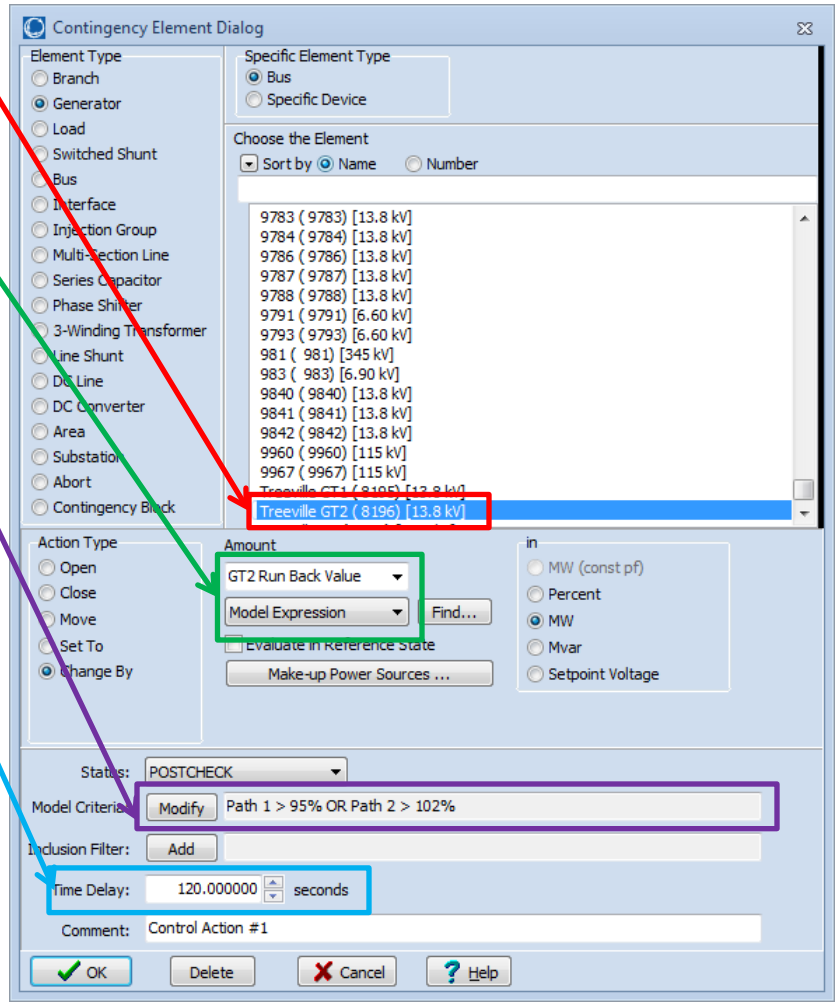
Treeville RAS Control Action #1: RemedialAction Gas Units

Choose Generator Unit 2

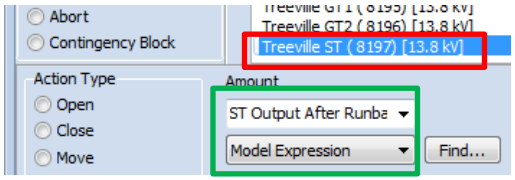
Model Expression for Unit 2

Model Criteria points to ModelFilter

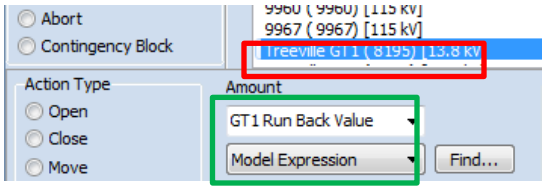
Time Delay



Same but for Steam Unit



Same but for Unit 1



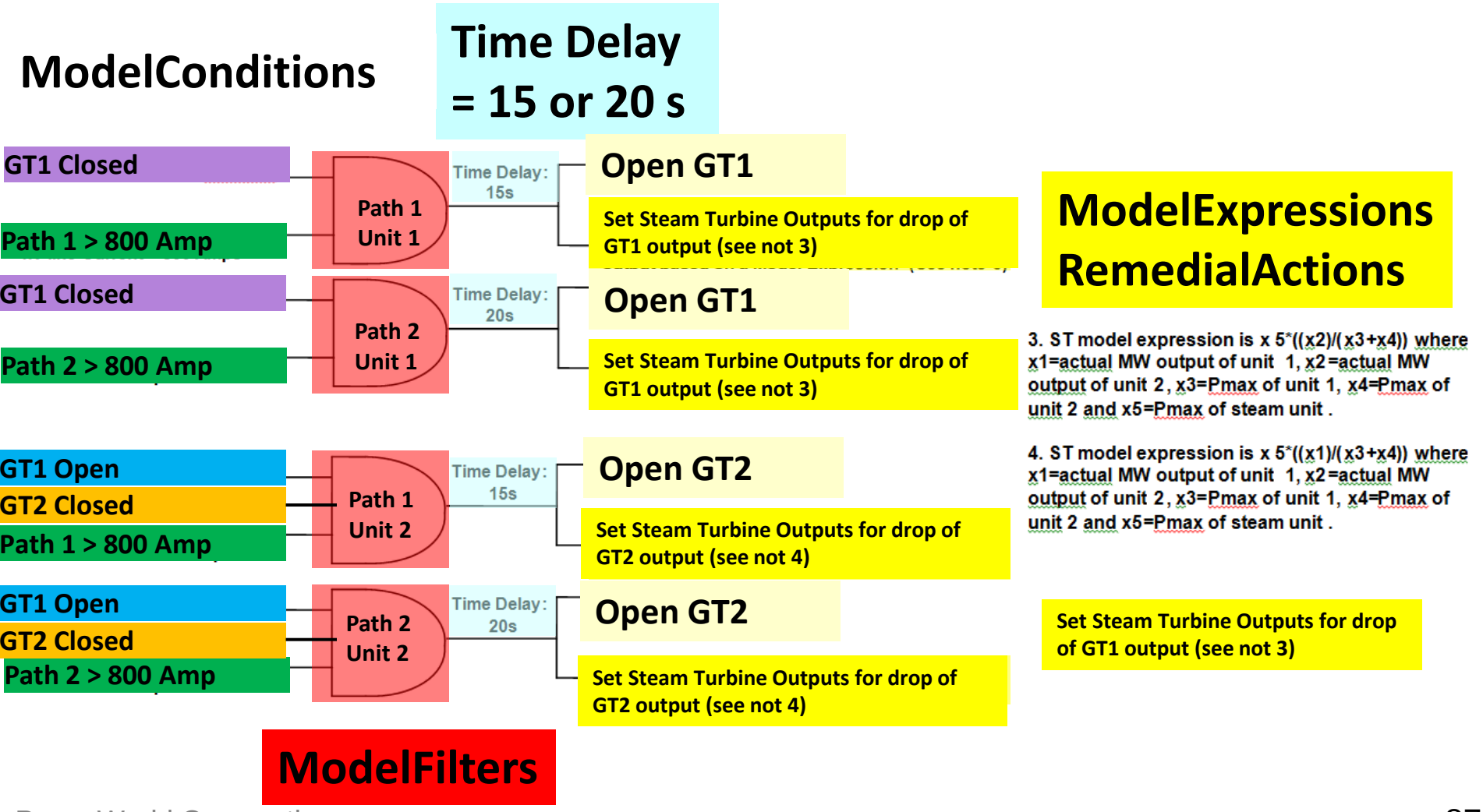
Treeville RAS Control Action #1: ModelExpression File Format

```
REMEDIALACTION (Name,Skip,Memo)
{
"Treeville Generation Run-Back Scheme" "NO " ""
}

REMEDIALACTIONELEMENT (RemedialAction,Object,Action,Criteria,
             CriteriaStatus,TimeDelay,InclusionFilter,Comment)
{
"Treeville Generation Run-Back Scheme" "GEN 8195" "CHANGEBY 'GT1 Run Back Value' MW"
               "Path 1 > 95% OR Path 2 > 102%" "POSTCHECK" 120.000000 "" "Control Action #1"

"Treeville Generation Run-Back Scheme" "GEN 8196" "CHANGEBY 'GT2 Run Back Value' MW"
               "Path 1 > 95% OR Path 2 > 102%" "POSTCHECK" 120.000000 "" "Control Action #1"

"Treeville Generation Run-Back Scheme" "GEN 8197" "SETTO 'ST Output After Runback' MW"
               "Path 1 > 95% OR Path 2 > 102%" "POSTCHECK" 120.000000 "" "Control Action #1"
}
```

Accept Questions

- Pause for questions from audience

Treeville RAS: Control Action #2



Treeville RAS Control Action #2: ModelCondition ModelFilter

```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,EvaluateInRef,DisableIfTrueInRef,Memo)
{
"GT1 Closed"     "Gen '8195' '1'"            "Gen" "AND" "NO" "NO " ""
"GT1 Open"       "Gen '8195' '1'"            "Gen" "AND" "NO" "NO " ""
"GT2 Closed"     "Gen '8196' '2'"            "Gen" "AND" "NO" "NO " ""
"Path 1 > 800 A" "Branch '8222' '8194' '1'"  "Branch" "AND" "NO" "NO " ""
"Path 2 > 800 A" "Branch '10440' '8194' '1'" "Branch" "AND" "NO" "NO " ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,
   ObjectField,ConditionType,Value,OtherValue,Absolute)
{
"GT1 Closed"     1 "Status" "startswith" "C" "" "NO "
"GT1 Open"       1 "Status" "startswith" "O" "" "NO "
"GT2 Closed"     1 "Status" "startswith" "C" "" "NO "
"Path 1 > 800 A" 1 "AmpsMax" ">" "800" "" "NO "
"Path 2 > 800 A" 1 "AmpsMax" ">" "800" "" "NO "
}
```

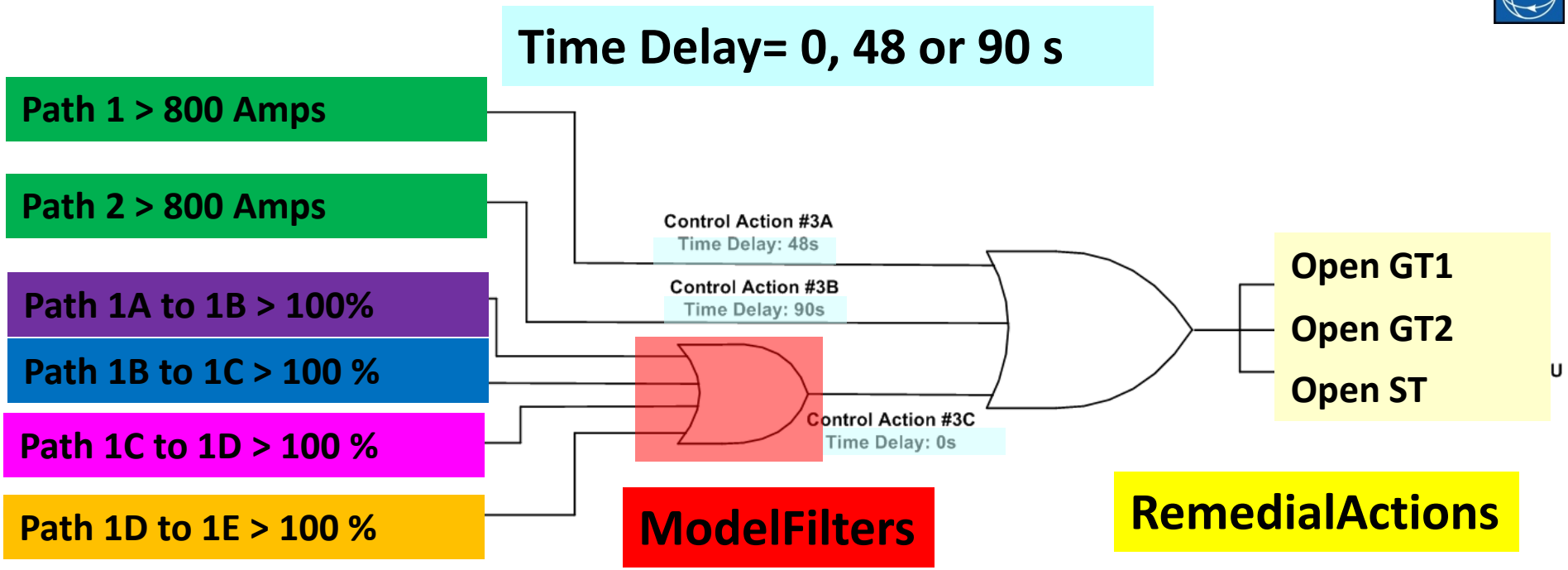
```
MODELFILTER (Name,Logic,Memo)
{
"Path 1 Unit 1" "AND" ""

"Path 2 Unit 1" "AND" ""

"Path 1 Unit 2" "AND" ""


"Path 2 Unit 2" "AND" ""
}
```

```
MODELFILTERCONDITION (ModelFilter,CondNum,Criteria,Logic)
{
"Path 1 Unit 1" 1 "GT1 Closed"     ""
"Path 1 Unit 1" 2 "Path 1 > 800 A" ""
"Path 2 Unit 1" 1 "GT1 Closed"     ""
"Path 2 Unit 1" 2 "Path 2 > 800 A" ""
"Path 1 Unit 2" 1 "GT1 Open"       ""
"Path 1 Unit 2" 2 "GT2 Closed"     ""
"Path 1 Unit 2" 3 "Path 1 > 800 A" ""
"Path 2 Unit 2" 1 "GT1 Open"       ""
"Path 2 Unit 2" 2 "GT2 Closed"     ""
"Path 2 Unit 2" 3 "Path 2 > 800 A" ""
}
```

Treeville RAS Control Action #2: ModelExpression, RemedialAction

```
MODELEXPRESSION (Name,Type,Expression,Memo,Object1,x1,BlankZero1,Object2,x2,BlankZero2,Object3,x3,
   BlankZero3,Object4,x4,BlankZero4,Object5,x5,BlankZero5,Object6,x6,BlankZero6,
   Object7,x7,BlankZero7,Object8,x8,BlankZero8)
{
"ST Output for GT1 Outage" "Expression" "x5*((x2)/(x3+x4))" ""
                "Gen '8195' '1'" "MW"    "YES" "Gen '8196' '2'" "MW"    "YES"
                "Gen '8195' '1'" "MWMax" "YES" "Gen '8196' '2'" "MWMax" "YES"
                "Gen '8197' 'L'" "MWMax" "YES" "" "" "NO " "" "" "NO " "" "" "NO "
"ST Output for GT2 Outage" "Expression" "x5*((x1)/(x3+x4))" ""
                "Gen '8195' '1'" "MW"    "YES" "Gen '8196' '2'" "MW“    "YES"
                "Gen '8195' '1'" "MWMax" "YES" "Gen '8196' '2'" "MWMax" "YES"
                "Gen '8197' 'L'" "MWMax" "YES" "" "" "NO " "" "" "NO " "" "" "NO "
}
REMEDIALACTION (Name,Skip,Memo)
{
"Treeville Generation Run-Back Scheme" "NO " ""
}
REMEDIALACTIONELEMENT (RemedialAction,Object,Action,Criteria,CriteriaStatus,TimeDelay,InclusionFilter,
   Comment)
{
"Treeville Generation Run-Back Scheme" "GEN 8195 1" "OPEN"
           "Path 1 Unit 1" "POSTCHECK" 15.000000 "" "Control Action #2A"
"Treeville Generation Run-Back Scheme" "GEN 8196 2" "OPEN"
           "Path 1 Unit 1" "POSTCHECK" 15.000000 "" "Control Action #2A"
"Treeville Generation Run-Back Scheme" "GEN 8195 1" "OPEN"
           "Path 2 Unit 1" "POSTCHECK" 20.000000 "" "Control Action #2B"
"Treeville Generation Run-Back Scheme" "GEN 8196 2" "OPEN"
           "Path 2 Unit 1" "POSTCHECK" 20.000000 "" "Control Action #2B“
"Treeville Generation Run-Back Scheme" "GEN 8197"   "SETTO 'ST Output for GT1 Outage' MW"
           "Path 1 Unit 2" "POSTCHECK" 15.000000 "" "Control Action #2A"
"Treeville Generation Run-Back Scheme" "GEN 8197"   "SETTO 'ST Output for GT2 Outage' MW"
           "Path 1 Unit 2" "POSTCHECK" 15.000000 "" "Control Action #2A"
"Treeville Generation Run-Back Scheme" "GEN 8197"   "SETTO 'ST Output for GT1 Outage' MW"
           "Path 2 Unit 2" "POSTCHECK" 20.000000 "" "Control Action #2B"
"Treeville Generation Run-Back Scheme" "GEN 8197"   "SETTO 'ST Output for GT2 Outage' MW"
           "Path 2 Unit 2" "POSTCHECK" 20.000000 "" "Control Action #2B"
}
```

Treeville RAS: Control Action #3



Treeville RAS Control Action #2: ModelCondition ModelFilter

```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,EvaluateInRef,DisableIfTrueInRef,Memo)
{
"Path 1 > 800 A"        "Branch '8222' '8194' '1'" "Branch" "AND" "NO" "NO " ""
"Path 2 > 800 A"        "Branch '10440' '8194' '1'" "Branch" "AND" "NO" "NO " ""
"Path 1A to 1B > 100%" "Branch '8222' '8226' '1'" "Branch" "AND" "NO" "NO " ""
"Path 1B to 1C > 100%" "Branch '8215' '8226' '1'" "Branch" "AND" "NO" " NO " ""
"Path 1C to 1D > 100%" "Branch '8201' '8215' '1'" "Branch" "AND" "NO" "NO " ""
"Path 1D to 1E > 100%" "Branch '8201' '8223' '1'" "Branch" "AND" "NO" "NO " ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,
   ObjectField,ConditionType,Value,OtherValue,Absolute)
{
"Path 1 > 800 A" 1 "AmpsMax" ">" "800" "" "NO "
"Path 2 > 800 A" 1 "AmpsMax" ">" "800" "" "NO "
"Path 1A to 1B > 100%" 1 "Percent" ">" "100" "" "NO "
"Path 1B to 1C > 100%" 1 "Percent" ">=" "100" "" "NO "
"Path 1C to 1D > 100%" 1 "Percent" ">=" "100" "" "NO "
"Path 1D to 1E > 100%" 1 "Percent" ">=" "100" "" "NO "
}
```

```
MODELFILTER (Name,Logic,Memo)
{
"Path 1 > 100%" "OR" ""
}
MODELFILTERCONDITION
(ModelFilter,CondNum,Criteria,Logic)
{
"Path 1 > 100%" 1 "Path 1A to 1B > 100%" ""
"Path 1 > 100%" 2 "Path 1B to 1C > 100%" ""
"Path 1 > 100%" 3 "Path 1C to 1D > 100%" ""
"Path 1 > 100%" 4 "Path 1D to 1E > 100%" ""
}
```

Treeville RAS Control Action #2: ModelExpression, RemedialAction

```
REMEDIALACTION (Name,Skip,Memo)
{
"Treeville Generation Run-Back Scheme" "NO " ""
}
REMEDIALACTIONELEMENT (RemedialAction,Object,Action,
          Criteria,CriteriaStatus,TimeDelay,InclusionFilter,Comment)
{
// Trip GT1
"Treeville Generation Run-Back Scheme" "BRANCH 8195 8194 1" "OPEN" "Path 1 > 800 A"
          "POSTCHECK" 90.000000 "" "Control Action #3B"
"Treeville Generation Run-Back Scheme" "BRANCH 8195 8194 1" "OPEN" "Path 2 > 800 A"
          "POSTCHECK" 48.000000 "" "Control Action #3A"
"Treeville Generation Run-Back Scheme" "BRANCH 8195 8194 1" "OPEN" "Path 1 > 100%"
          "POSTCHECK" 0         "" "Control Action #3C"
// Trip GT2
"Treeville Generation Run-Back Scheme" "BRANCH 8196 8194 1" "OPEN" "Path 1 > 800 A"
          "POSTCHECK" 90.000000 "" "Control Action #3B"
"Treeville Generation Run-Back Scheme" "BRANCH 8196 8194 1" "OPEN" "Path 2 > 800 A"
          "POSTCHECK" 48.000000 "" "Control Action #3A"
"Treeville Generation Run-Back Scheme" "BRANCH 8196 8194 1" "OPEN" "Path 1 > 100%"
          "POSTCHECK" 0         "" "Control Action #3C"
// Trip ST
"Treeville Generation Run-Back Scheme" "BRANCH 8197 8194 1" "OPEN" "Path 1 > 800 A"
          "POSTCHECK" 90.000000 "" "Control Action #3B"
"Treeville Generation Run-Back Scheme" "BRANCH 8197 8194 1" "OPEN" "Path 2 > 800 A"
          "POSTCHECK" 48.000000 "" "Control Action #3A"
"Treeville Generation Run-Back Scheme" "BRANCH 8197 8194 1" "OPEN" "Path 1 > 100%"
          "POSTCHECK" 0         "" "Control Action #3C"
}
```

Treeville RAS: Note On Time Delay Parameter

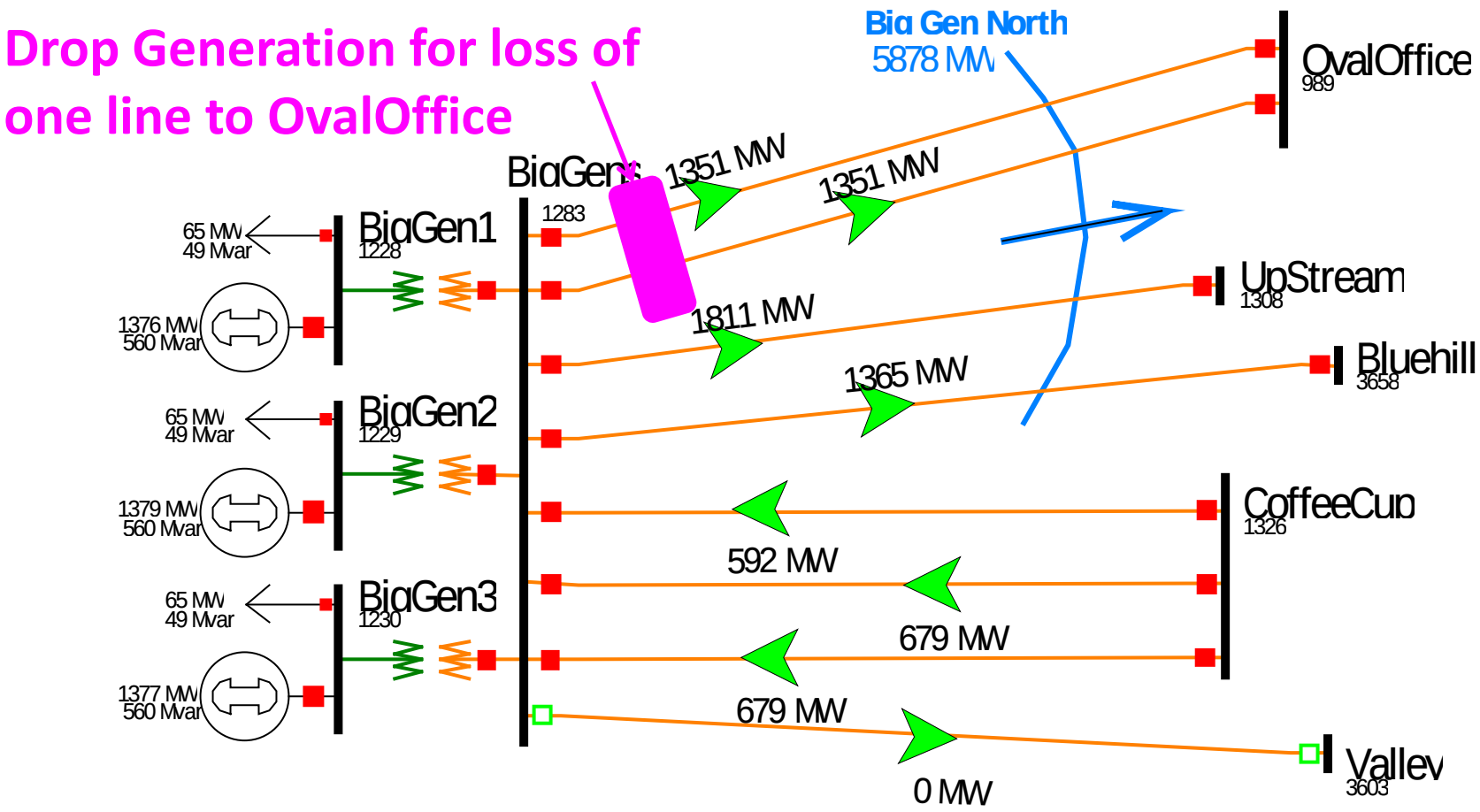
- For the Treeville RAS actions, there are time-delays assigned which are *important*
- Time Delays are associated with monitoring current on two branches leaving Treeville
  - Action #1: 120 seconds for 102% overload
  - Action #2: 15 or 20 seconds for over 800 Amps then trip only one gas plant (and modify steam)
  - Action #3: 48 or 90 seconds for over 800 Amps then trip both gas plants and steam plants
- Need to do Action #2 *first* which trips only one generator
  - That may make unnecessary Action #3 which trips all the generators at Treeville
- Details Matter!

Accept Questions

- Pause for questions from audience

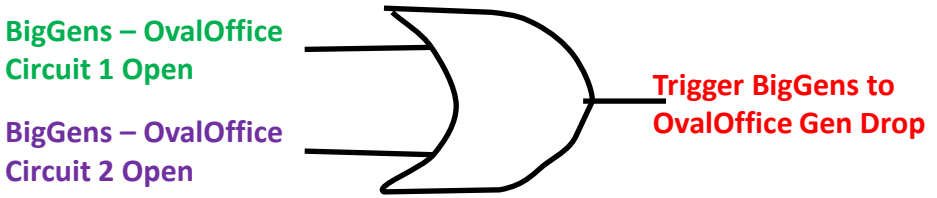
Example #2: Double Line Outage with Generation Dropping

- Consider system below



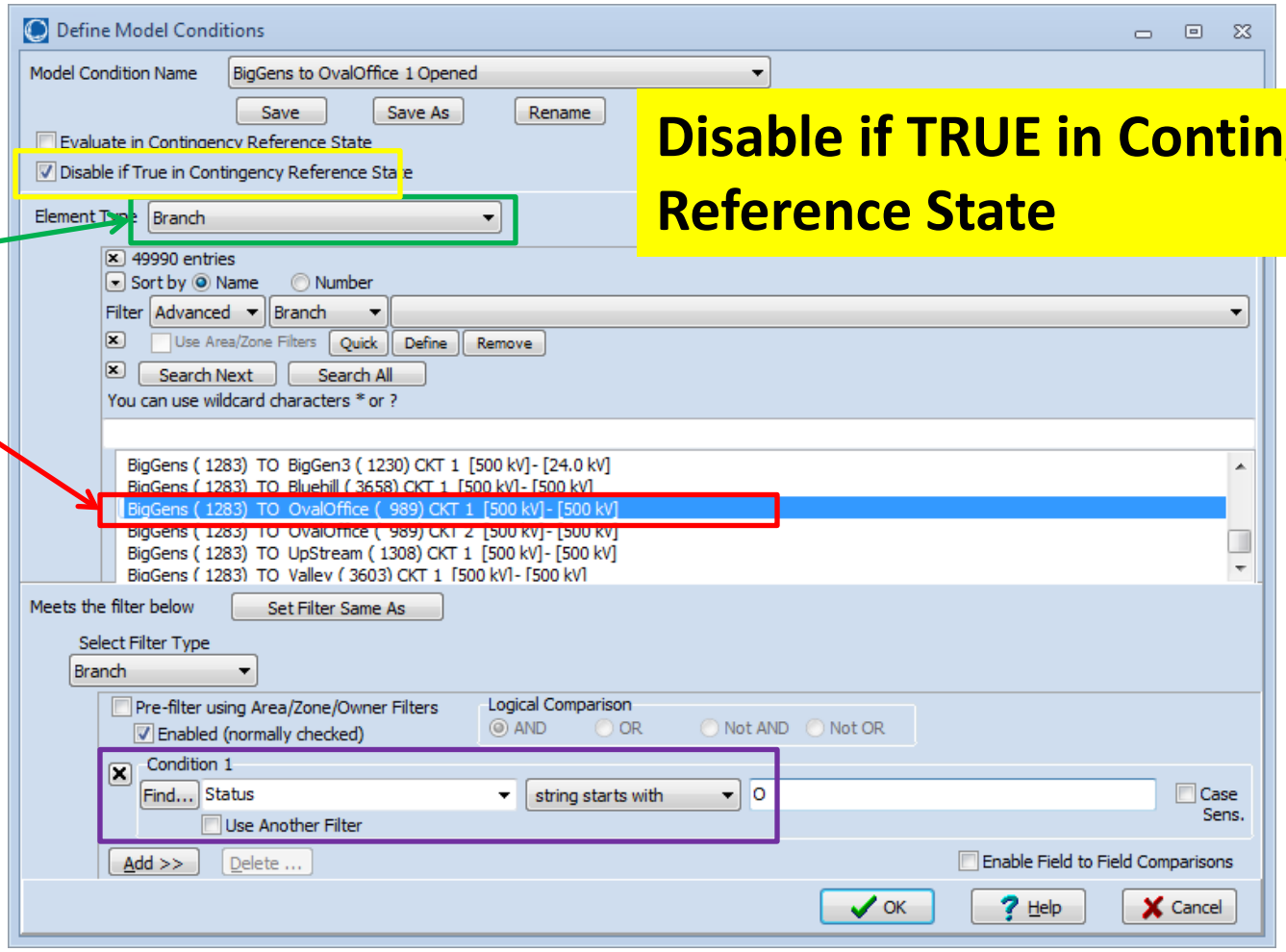
BigGens to OvalOffice Gen Drop

- RAS is triggered when either of lines from BigGens to OvalOffice are opened by the contingency
  - If one line is out in the initial system (for example for maintenance), the RAS will also trigger if the second line is opened during the contingency
- When the RAS is triggered, it will trip generation from the 3 BigGens units based on a lookup table which is a function of MW flow on the interface "Big Gen North"



Big Gen North MW Flow	Generation Change
0	0
3000	-500
4000	-1200
5000	-2000

BigGens to OvalOffice Gen Drop: ModelCondition Dialog



Disable if TRUE in Contingency Reference State

Type = Branch

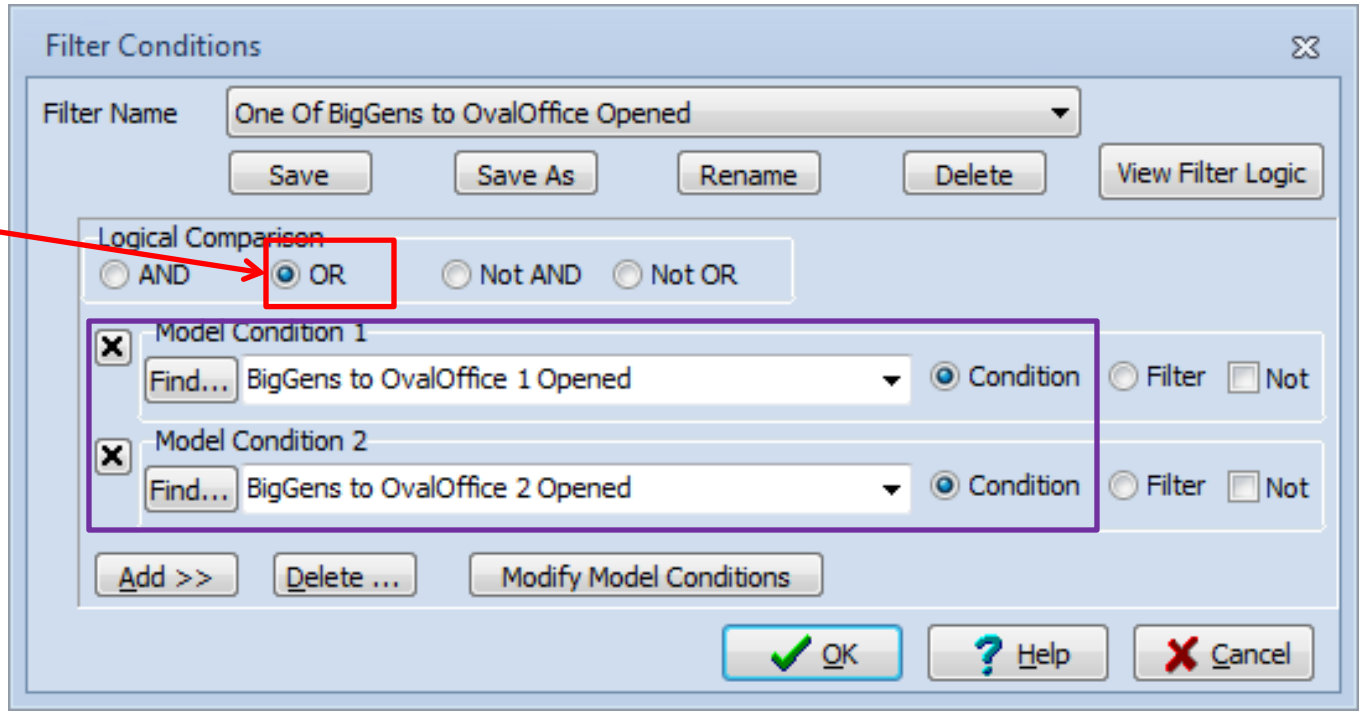
Choose Branch

Condition for Branch

BigGens to OvalOffice Gen Drop: ModelFilter Dialog

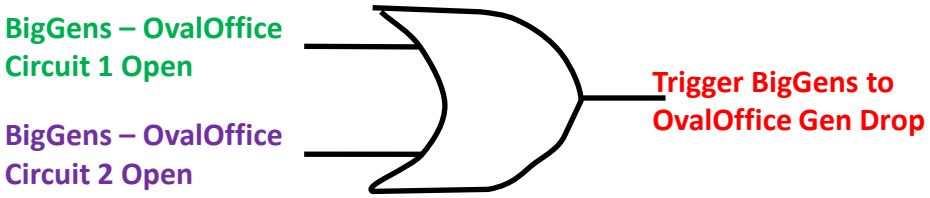
OR Logic

List of Model Conditions

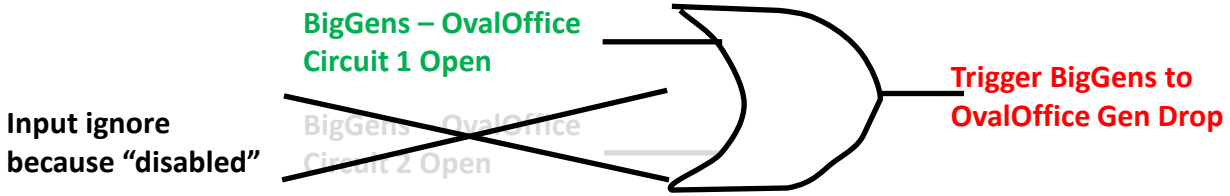


Disable if TRUE in Contingency Reference State Meaning

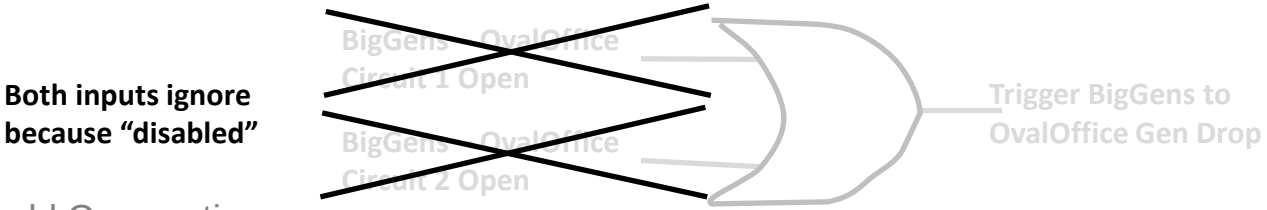
- If both lines are inservice then logic looks like



- If only circuit 1 is inservice then logic changes to



- If both lines are out of service then all inputs to the model filter are all “disabled” and thus it’s output is considered “Disabled”
  - If the model filter is fed into another filter that input is disabled
  - If model filter is used directly, it returns FALSE



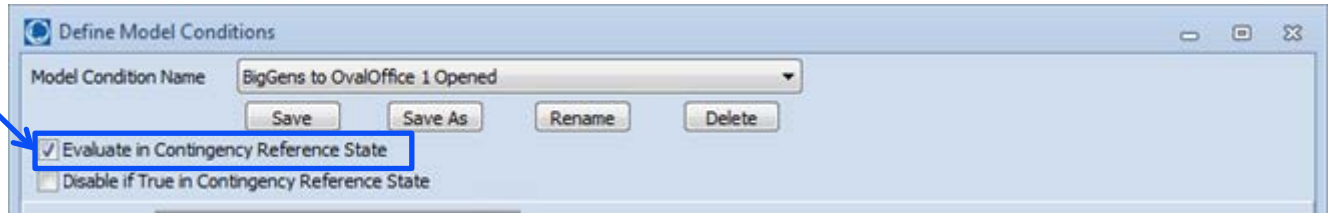
BigGens to OvalOffice Gen Drop: ModelCondition, ModelFilter File

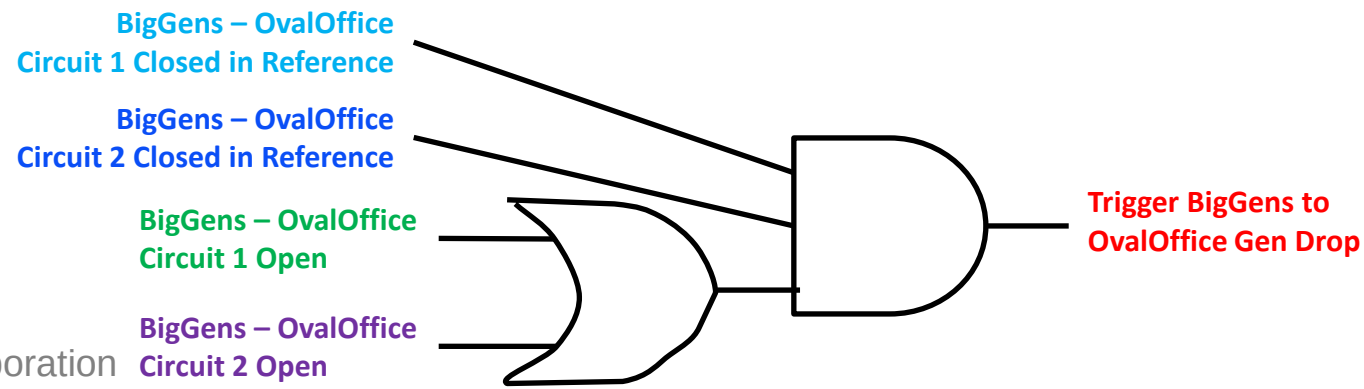
```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,EvaluateInRef,DisableIfTrueInRef,Memo)
{
"BigGens to OvalOffice 1 Opened" "Branch '1283' '989' '1'" "Branch" "AND" "NO" "YES" ""
"BigGens to OvalOffice 2 Opened" "Branch '1283' '989' '2'" "Branch" "AND" "NO" "YES" ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,ObjectField,ConditionType,Value,
                         OtherValue,Absolute)
{
"BigGens to OvalOffice 1 Opened" 1 "Status" "startswith" "O" "" "NO "
"BigGens to OvalOffice 2 Opened" 1 "Status" "startswith" "O" "" "NO "
}
MODELFILTER (Name,Logic,Memo)
{
"One Of BigGens to OvalOffice Opened" "OR" ""
}
MODELFILTERCONDITION (ModelFilter,CondNum,Criteria,Logic)
{
"One Of BigGens to OvalOffice Opened" 1 "BigGens to OvalOffice 1 Opened" ""
"One Of BigGens to OvalOffice Opened" 2 "BigGens to OvalOffice 2 Opened" ""
}
```

Different Logic for Reference State Outage

- What if RAS should only be triggered if both lines are in service initially?
  - Thus if one was out in the reference state you would not trigger the RAS

Evaluate in Contingency Reference State





Different Logic for Reference State Outage: File Format

```
MODELCONDITION (Name,Object,FilterObjectType,FilterLogic,EvaluateInRef,DisableIfTrueInRef,Memo)
{
"BigGens to OvalOffice 1 ClosedRef" "Branch '1283' '989' '1'" "Branch" "AND" "YES" "NO"  ""
"BigGens to OvalOffice 2 ClosedRef" "Branch '1283' '989' '2'" "Branch" "AND" "YES" "NO"  ""
"BigGens to OvalOffice 1 Opened"    "Branch '1283' '989' '1'" "Branch" "AND" "NO"  "YES" ""
"BigGens to OvalOffice 2 Opened"    "Branch '1283' '989' '2'" "Branch" "AND" "NO"  "YES" ""
}
MODELCONDITIONCONDITION (ModelCondition,CondNum,ObjectField,ConditionType,Value,
                         OtherValue,Absolute)
{
"BigGens to OvalOffice 1 ClosedRef" 1 "Status" "startswith" "C" "" "NO "
"BigGens to OvalOffice 2 ClosedRef" 1 "Status" "startswith" "C" "" "NO "
"BigGens to OvalOffice 1 Opened"    1 "Status" "startswith" "O" "" "NO "
"BigGens to OvalOffice 2 Opened"    1 "Status" "startswith" "O" "" "NO "
}
MODELFILTER (Name,Logic,Memo)
{
"One Of BigGens to OvalOffice Opened" "OR" "“
"NewFilterName" "OR" ""
}
MODELFILTERCONDITION (ModelFilter,CondNum,Criteria,Logic)
{
"One Of BigGens to OvalOffice Opened" 1 "BigGens to OvalOffice 1 Opened" ""
"One Of BigGens to OvalOffice Opened" 1 "BigGens to OvalOffice 2 Opened" ""
"NewFilterName"                       1 "BigGens to OvalOffice 1 ClosedRef" ""
"NewFilterName"                       2 "BigGens to OvalOffice 2 ClosedRef" ""
"NewFilterName"                       3 "One Of BigGens to OvalOffice Opened" ""
}
```

Accept Questions

- Pause for questions from audience

Implementing Gen Drop: Lookup Table,Model Expression

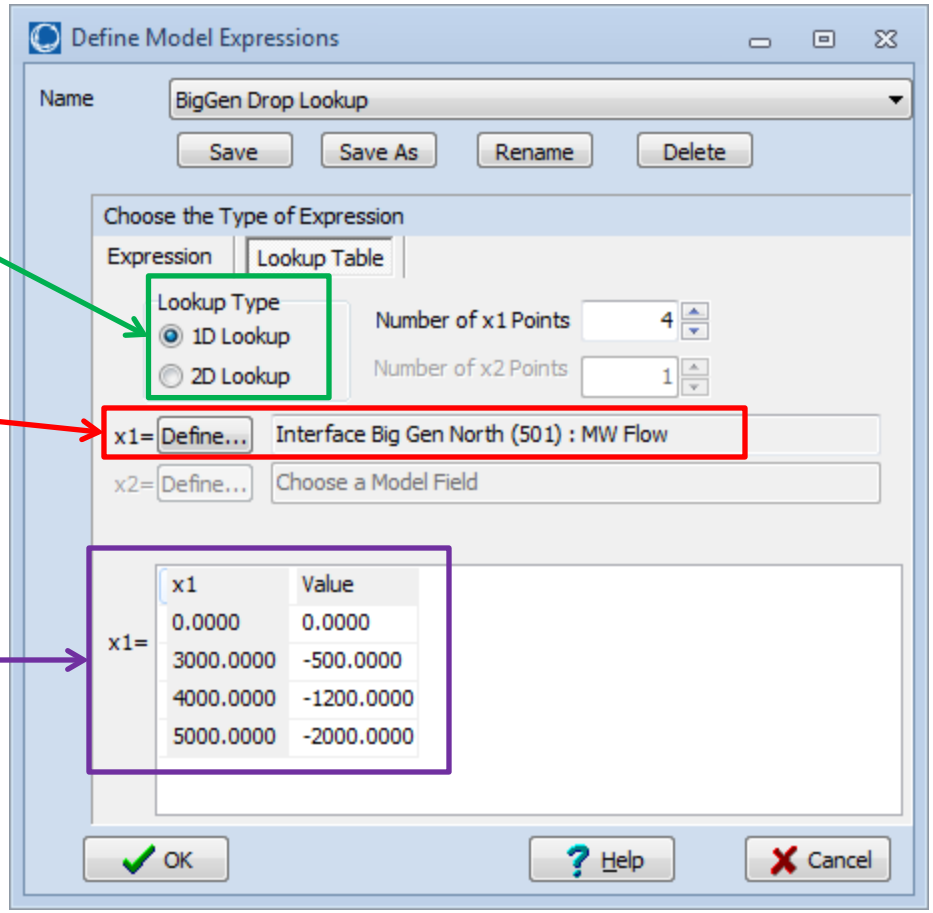
- 1D Lookup Table using a ModelExpression

Choose 1D or 2D lookup

X1 = MW Flow on Interface Big Gen North

Big Gen North MW Flow	Generation Change
0	0
3000	-500
4000	-1200
5000	-2000

Lookup Table

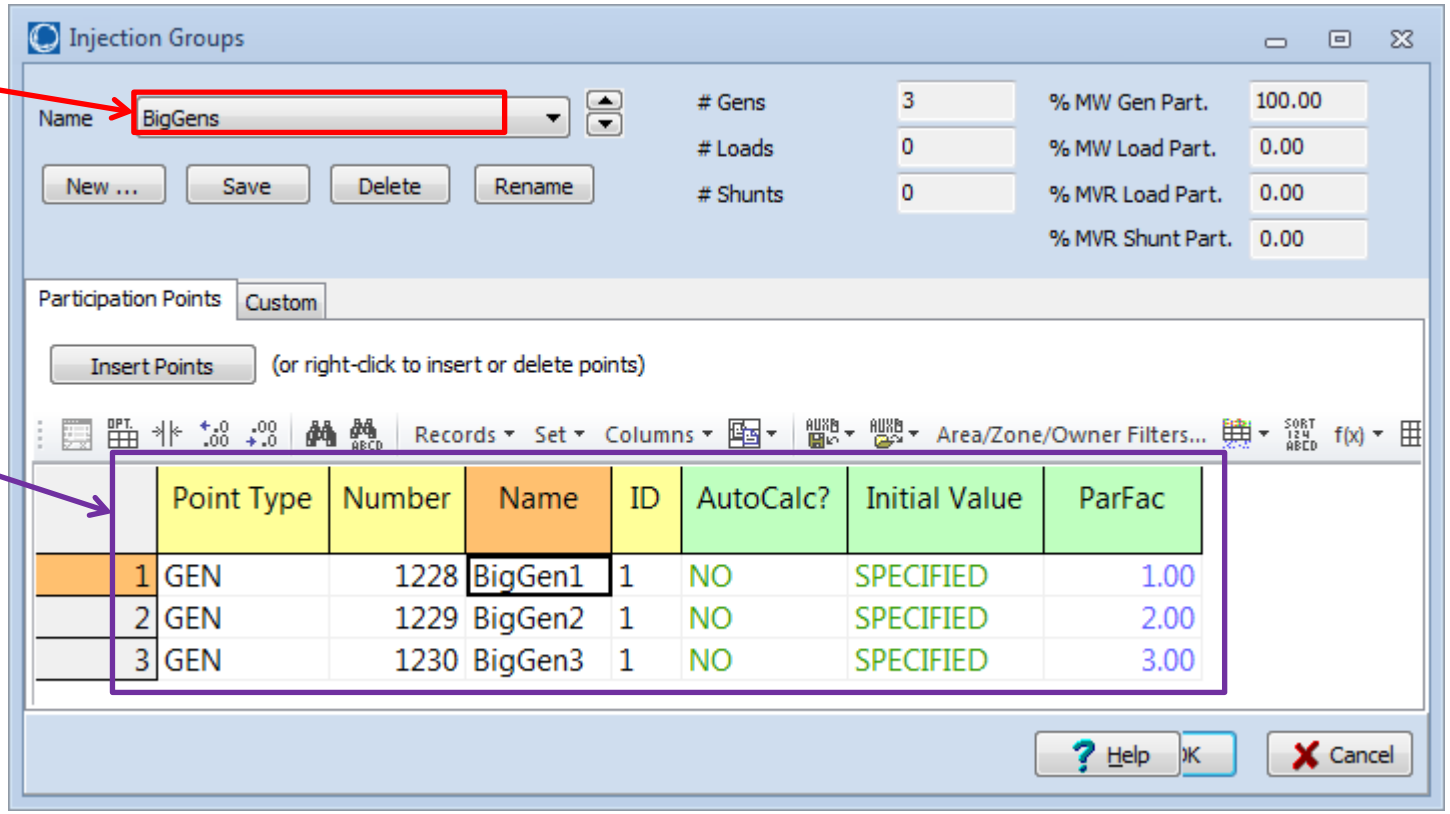


Injection Group Load Dropping

- First you must define an Injection Group

Name

List of Participation Points



BigGens to OvalOffice Gen Drop: ModelExpression, InjectionGroup

```
MODELEXPRESSION (Name,Type,Expression,Memo,
   Object1,x1,BlankZero1,Object2,x2,BlankZero2,Object3,x3,BlankZero3,
   Object4,x4,BlankZero4,Object5,x5,BlankZero5,Object6,x6,BlankZero6,
   Object7,x7,BlankZero7,Object8,x8,BlankZero8)
{
"BigGen Drop Lookup" "Lookup" "" ""
  "Interface 'Big Gen North'" "MW" "YES"
   "" "" "NO " "" "" "NO " "" "" "NO " "" "" "NO "
   "" "" "NO " "" "" "NO " "" "" "NO "
   <SUBDATA LookupTable>
     x1           value
        0.0         0.0
     3000.0      -500.0
     4000.0     -1200.0
     5000.0     -2000.0
   </SUBDATA>
}
```

Model Expression

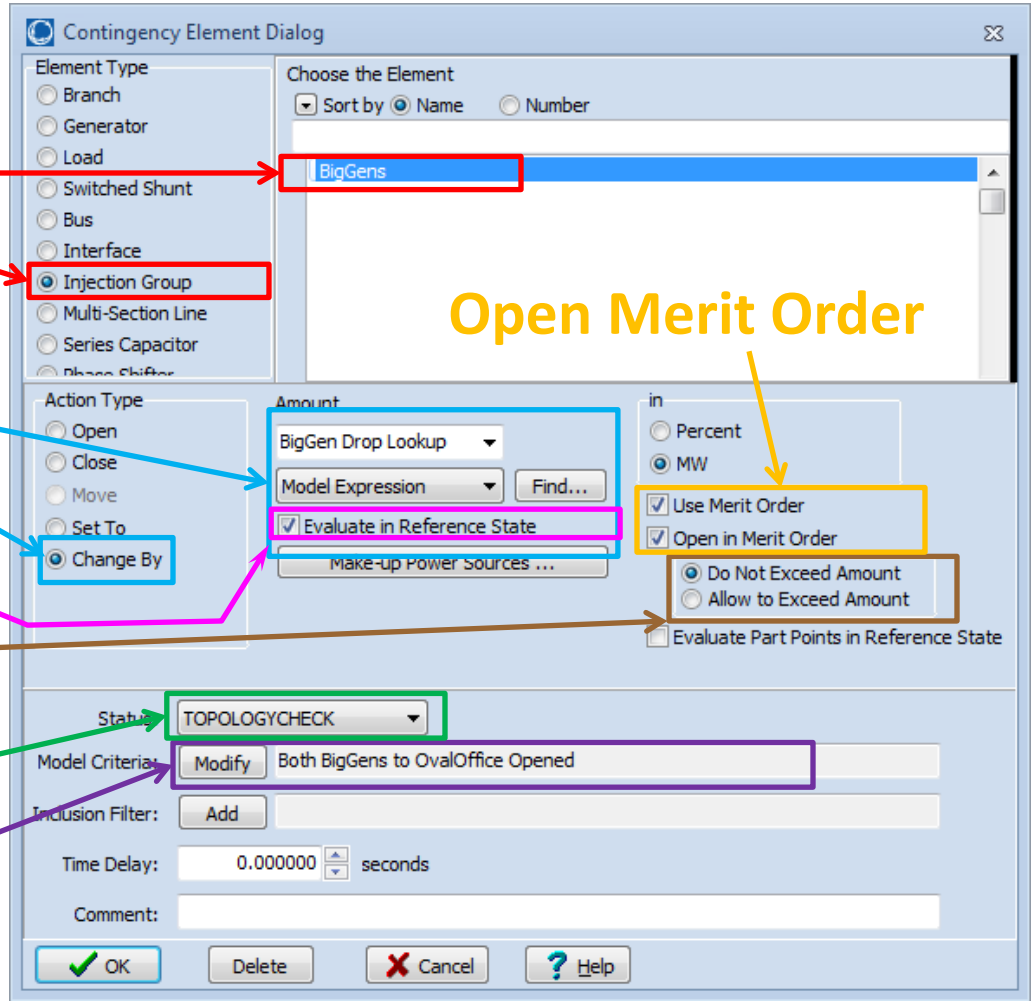
```
INJECTIONGROUP (Name)
{
"BigGens"
}
PARTPOINT (GroupName,Object,AutoCalcMethod,PartFact,AutoCalc)
{
"BigGens" "Gen '1228' '1'" "SPECIFIED" 1.00000 "NO "
"BigGens" "Gen '1229' '1'" "SPECIFIED" 2.00000 "NO "
"BigGens" "Gen '1230' '1'" "SPECIFIED" 3.00000 "NO "
}
```

Injection Group

Details, Details, Details: Gen Tripping using Lookup Table

- RAS will refer to an Injection Group to do generation (or load) tripping
  - Tripping is done in the order of highest participation factor as assigned by PartPoints in injection group
- Trip based on Lookup table… Details matter
  - When should lookup calculation be done?
    - Often "gen drop arming levels" are based on system conditions before any event happens
    - Must base calculation on the Reference State as the interface flows may change during the contingency solution process
  - Tripping is done one unit at a time
    - Ordering is done using an injection group using "Merit Order"
  - Choice
    - Do not EXCEED amount of lookup table
    - Allow to EXCEED amount
- In this example it may be that the double line outage results in unsolvable power flow solution → TOPOLOGYCHECK

BigGens to OvalOffice Gen Drop: RemedialAction



BigGens to OvalOffice Gen Drop: ModelExpression, InjectionGroup

```
REMEDIALACTION (Name,Skip,Memo)
{
"BigGen OvalOffice Gen Drop" "NO " ""
}

REMEDIALACTIONELEMENT (RemedialAction,Object,Action,
 Criteria,CriteriaStatus,TimeDelay,InclusionFilter,Comment)
{
"BigGen OvalOffice Gen Drop"
"INJECTIONGROUP 'BigGens'"
"CHANGEBY 'BigGen Drop Lookup' MWMERITORDEROPEN REF"
"Both BigGens to OvalOffice Opened" "TOPOLOGYCHECK" 0 "" ""
}
```

TOPOLOGYCHECK

Criteria based on *Status Only*

- Persistent problem in contingency runs when using RAS
  - RAS is configured to open 2,000 MW of generation when Line X trips
  - Before 2012, this was achieved by configuring a *POSTCHECK* action that looks at the branch status and trips generation if the branch is out of service
  - Works most of the time, but...
  - What if the outage of Line X results in an unsolvable steady state power flow solution?
    - Basically means that the RAS scheme is actually preventing a voltage collapse from occurring
    - This means that the *POSTCHECK* action is never evaluated because the power flow solution failed.

- Solution: *TOPOLOGYCHECK* actions

Summary

- Communication
  - Develop relationships with folks in your company who maintain RAS and Relays
  - Communicate across utilities and WECC members
- Fundamental pieces of RAS are simple
  - But… the details of how they are put together are very particular
  - And… these details matter
- RAS file format structure is being maintained *incrementally* by PowerWorld
  - http://www.powerworld.com/files/PowerWorld_RASFileFormat.pdf
- Come to the workshop on May 27 – 29, 2015

Come to the Workshop on May 27 – 29, 2015

- In-Person Workshop
  - May 27 – 29, 2015
  - WECC offices in Salt Lake City
  - Starts 1 PM May 27, Ends at Noon May 29
  - There will be no WebEx for this meeting. In-Person Only.

Monday 25	Tuesday 26	Wednesday 27	Thursday 28	Friday 29
		Classroom Style / 1 PM	Break-Out Groups	Noon

Classroom Style and Break-Out Groups

- Classroom Style Portion
  - Presentations on how to implement the various input parts that represent RAS and Relay models
    - Contingency Actions
    - Boolean Logic
    - Lookup tables and Expressions
  - Bring you laptop and work along with us
- Break-out Groups Portion
  - There will be several engineers from throughout WECC with experience implementing their RAS and Relay Models working attending this workshop
  - We will break the attendees into smaller groups to spend time implementing their actual RAS
  - *Bring your RAS descriptions* so you can work on them
  - Bring your laptop and go home with some real RAS modeled
- Entire workshop is hands-on → There will be no WebEx/Phone

Accept Questions

- Any more questions to finish