Abstract

This document describes the Grid Research for Good (GRG) schema to represent static transmission networks.

1 Basic Concepts

This section defines the basic concepts used to model grid data. We first define data types for representing numerical values, variables, and object statuses. Then we define data types used to represent electrical values.

Throughout the document we adopt the symbol \( j \) to denote the complex imaginary unit, which satisfies the equation: \( j^2 = -1 \). For a complex number \( x \in \mathbb{C} \) we write \( \bar{x} \) to denote the conjugate of \( x \). Additionally, for a list of elements \( A = (a_1, \ldots, a_n) \), we use the notation \( A[k] \) to indicate the \( k^{th} \) element of \( A \), \( a_k \).

For a JSON attribute, the keyword \$\text{ref} \) describes a JSON Reference, i.e., a reference to an object whose fragment part is a URI encoded JSON Pointer. In this document, references are local to the schema, and their values reflect the document hierarchy. To allow ease of format extension in the future and ease of format conversion between GRG and other file formats in the power system community, we set ‘additionalProperties’ to ‘true’ in our schema to allow users define additional properties.

1.1 Values

Extended Number

An extended number is a numeric data type whose value is either a number or one of the following: \( \text{Inf} \), denoting \( \infty \), \( -\text{Inf} \) denoting \( -\infty \), \( \text{NaN} \), standing for “not a number” and representing an undefined or unrepresentable value, or \( \text{Null} \), to denote a value which is missing.

```json
"extended_number": {
  "oneOf": [
    {"type": "number"},
    {"enum": ["Inf", "-Inf", "NaN", "Null"]}
  ]
}
```

The following example illustrates the above concept by where \( x \) is assigned a numeric value, and \( y \) to a \( -\infty \).

```json
"x": 10.23,
"y": "-Inf"
```

*Contacts:* Terrence W.K. Mak: \texttt{wmak@umich.edu}, Pascal Van Hentenryck: \texttt{pvanhent@umich.edu}
Extended Positive Number

An extended positive number restricts the notion of extended number to non-negative values and the special values Inf, NaN, and Null.

```json
"extended_positive_number": {
  "oneOf": [
    {"type": "number", "minimum": 0},
    {"enum": ["Inf", "NaN", "Null"]}
  ]
}
```

Domain

A domain is a collection of values used to describe valid assignments for a variable. We define three types of domains:
- Finite domains describing a collection of strings.
- Finite domains describing a collection of numbers.
- Bound domains describing a range \([lb, ub] \subseteq \mathbb{R}\).

```json
"domain": {
  "type": "object",
  "required": ["var"],
  "properties": {
    "var": {
      "oneOf": [
        {"type": "array", "items": {"type": "string"}, "minItems": 1},
        {"type": "array", "items": {"type": "number"}, "minItems": 1},
        {"type": "object", "required": ["lb", "ub"],
          "properties": {
            "lb": {$ref: "#/values/extended_number"},
            "ub": {$ref: "#/values/extended_number"}
          },
          "additionalProperties": true
        }
      ],
      "additionalProperties": true
    }
  }
}
```

Following are examples of finite string domain (x), finite numerical domain (y), and bound domain (z).

```json
"x": { "var": ["a", "b", "c"] },
"y": { "var": [1, 2, 3] },
"z": { "var": {"lb": 0, "ub": 10} }
```

Positive Domain

A positive domain restricts the notion of domain to non-negative values.

```json
"positive_domain": {
  "type": "object",
  "required": ["var"],
  "properties": {
    "var": {
      "oneOf": [
        {"type": "number", "minimum": 0},
        {"enum": ["Inf", "NaN", "Null"]}
      ]
    }
  }
}
```
Abstract Value

An abstract value is an extended numeric data type which can describe either a numeric value or a variable.

Example: abstract value

```
"x": { "var": { "lb": 0, "ub": 10 } },
"y": 3.56
```

Abstract Positive Value

An abstract positive value restricts the notion of abstract value to non-negative numeric values and non-negative variables.

```
"x": { "var": { "lb": 0, "ub": 10 } },
"y": 3.56
```

Status

A status is a special boolean variable whose domain elements are “on” and “off”.

```
"status": { "oneOf": [ { "enum" : [ "off", "on" ]},
```
In the following examples, \( x \) represents an unassigned status variable, and \( y \) represents a status element whose value is ‘on’.

\[
\begin{align*}
&x: \quad \{ \text{ "var": } [\text{ "on", } \text{ "off"} ] \} \\
&y: \quad \text{ "on"},
\end{align*}
\]

**GRG Pointer**

A *GRG pointer* is a string used to identify an object’s value in the GRG document. A GRG pointer extends a JSON pointer by adopting the following prefixes:

- \( # \), which refers to the document root.
- \( @ \), which refers to a JSON object in the same scope as the pointer itself.

If the pointer does not start with either \( # \) or \( @ \), we assume the pointer refers to a component by its unique ID.

\[
\begin{align*}
&\text{"grg_pointer": } \{ \\
&\quad \text{ "type": } \text{ "string"}, \\
&\quad \text{ "pattern": } \text{ ".*"} \\
&\} 
\end{align*}
\]

In the following examples, the GRG pointers \( p1 \), \( p2 \), and \( p3 \) refer to the same value (\text{voltage\_id\_ALH\_2}). The GRG pointer \( p3 \) uses the global GRG id of the component object for referencing. In GRG format, we assume all GRG ids (id) are global and unique for all the components (i.e. GRG ids are global identifiers of components). The pointer \( p4 \) refers to the first value of the array \text{ var } in the object \text{ status } of \text{ switch\_example }.

\[
\begin{align*}
&\text{"network": } \{ \\
&\quad \text{ "components": } \{ \\
&\quad \quad \text{ "switch\_example": } \{ \\
&\quad \quad \quad \text{ "type": } \text{ "switch"}, \\
&\quad \quad \quad \text{ "subtype": } \text{ "breaker"}, \\
&\quad \quad \quad \text{ "id": } \text{ "sw\_722"}, \\
&\quad \quad \quad \text{ "link\_1": } \text{ "voltage\_id\_ALH\_2"}, \\
&\quad \quad \quad \text{ "link\_2": } \text{ "voltage\_id\_ALH\_3"}, \\
&\quad \quad \quad \text{ "status": } \{ \text{ "var": } [\text{ "off", } \text{ "on"}], \\
&\quad \quad \quad \text{ "p1": } @/\text{link\_1} \\
&\quad \quad \} \\
&\quad \} \\
&\quad \},
\end{align*}
\]

\[
\begin{align*}
&p2: \quad #/\text{network/components/switch\_example/link\_1}, \\
&p3: \quad \text{ "sw\_722/\text{link\_1}"}, \\
&p4: \quad \text{ "sw\_722/status/var/0"
\end{align*}
\]
A table is an object linking a list of GRG elements \( \langle e_1, \ldots, e_k \rangle \) to a set of tuples \( \{ T_1, \ldots, T_n \} \), where each \( T_i = \langle v_1, \ldots, v_k \rangle \) has values \( v_i \) (\( i = 1, \ldots, k \)). The elements are referred to as table arguments, and the set of value tuples as table values. In other words, a table expresses the relation between a list of elements and the set of possible values for such elements.

```
"table": {
  "type" : "object",
  "required": ["arguments", "values"],
  "properties": {
    "arguments": {
      "type" : "array",
      "items" : {"$ref": "/#values/grg_pointer"},
      "minItems": 1, "uniqueItems": true
    },
    "values": {
      "type" : "array",
      "items": {
        "type" : "array",
        "items": {
          "oneOf": [
            {"type": "#/values/abstract_value"},
            {"type": "string"}
          ]
        },
        "minItems": 1
      },
      "minItems": 1
    }
  }
}
```

The following example describes three assignments for the elements \( x \), \( y \), and \( z \):

\[
\begin{align*}
  x &= 1, \quad y = 12, \quad z = 13 \\
  x &= 2, \quad y = 6, \quad z = 20 \\
  x &= 3, \quad y = 10, \quad z = 31
\end{align*}
\]

```
"table_1": {
  "arguments": ["#/x", "/#y", "/#z"],
  "values": [
    [1, 12, 13],
    [2, 6, 20],
    [3, 10, 31]
  ]
}
```

### 1.2 Electrical Values

This section defines the electrical values adopted by the GRG schema.

**Impedance**

Electrical impedance measures the opposition of a circuit to a current when a given voltage is applied. Impedance is represented as a complex quantity \( Z \):

\[
Z = R + jX,
\]

(1)

where \( R \) denotes the resistance, and \( X \) the reactance.

Table 1 maps the real and imaginary components of impedance to their GRG schema counterparts.
### Impedance

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>impedance</td>
<td>( Z )</td>
<td>Ohm ((\Omega))</td>
</tr>
<tr>
<td>resistance</td>
<td>( R )</td>
<td>Ohm ((\Omega))</td>
</tr>
<tr>
<td>reactance</td>
<td>( X )</td>
<td>Ohm ((\Omega))</td>
</tr>
</tbody>
</table>

Table 1: Impedance: representation in the GRG impedance element and units.

```json
{ "impedance": {  
  "type": "object",  
  "required": ["resistance", "reactance"],  
  "additionalProperties": true,  
  "properties": {  
    "resistance": {"$ref": "/values/abstract_value"},  
    "reactance": {"$ref": "/values/abstract_value"}  
  }  
}
```

Though the schema definition may seem complicated, the following example shows how simple it is to use:

```json
{  
  "impedance": {  
    "reactance": 6.52,  
    "resistance": 2.39  
  }  
}
```

### Admittance

Electrical admittance is a measure of how much a circuit allows current to flow. Admittance is represented by a complex quantity \( Y \):

\[
Y = G + jB, \tag{2}
\]

where \( G \) denotes conductance, and \( B \) denotes susceptance. These real and imaginary components are mapped to the GRG admittance object as shown in Table 2.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>admittance</td>
<td>( Y )</td>
<td>Siemens ((S))</td>
</tr>
<tr>
<td>conductance</td>
<td>( G )</td>
<td>Siemens ((S))</td>
</tr>
<tr>
<td>susceptance</td>
<td>( B )</td>
<td>Siemens ((S))</td>
</tr>
</tbody>
</table>

Table 2: Admittance: representation in the GRG admittance element and units.

```json
{ "admittance": {  
  "type": "object",  
  "required": ["conductance", "susceptance"],  
  "additionalProperties": true,  
  "properties": {  
    "conductance": {"$ref": "/values/abstract_value"},  
    "susceptance": {"$ref": "/values/abstract_value"}  
  }  
}
```

The following example shows how to define an admittance in GRG format:

```json
{  
  "shunt": {  
    "conductance": 0,  
    "susceptance": 2.3e-05  
  }  
}
```
**Power**

Electric power is the rate at which electrical energy is transferred by an electric circuit. We define the complex power $S$ as:

$$ S = P + jQ, $$

where $P$ is the active (or real) power, and $Q$ is the reactive power. Table 3 shows how these quantities are encoded in the GRG format.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>power</td>
<td>$S$</td>
<td>MegaWatt ($MW$)</td>
</tr>
<tr>
<td>active</td>
<td>$P$</td>
<td>MegaWatt ($MW$)</td>
</tr>
<tr>
<td>reactive</td>
<td>$Q$</td>
<td>MegaVolt-Ampere Reactive ($MVAR$)</td>
</tr>
</tbody>
</table>

Table 3: Power: representation in the GRG format and units.

Active and reactive power are defined as variables with bound domains in the following example:

```json
"power": {
  "type": "object",
  "required": ["active", "reactive"],
  "additionalProperties": true,
  "properties": {
    "active": {"$ref": "/#/values/abstract_value"},
    "reactive": {"$ref": "/#/values/abstract_value"}
  }
}
```

**Voltage**

Voltage is the difference in electric potential energy between two points per unit electric charge. We define the voltage phasor $V$ as:

$$ V = v \cdot e^{j\theta}, $$

where $v$ is the voltage magnitude, and $\theta$ is the voltage phase angle. Table 4 connects the phasor components and units to their GRG representations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltage</td>
<td>$V$</td>
<td>kiloVolt ($kV$)</td>
</tr>
<tr>
<td>magnitude</td>
<td>$v$</td>
<td>Degrees</td>
</tr>
<tr>
<td>angle</td>
<td>$\theta$</td>
<td>Degrees</td>
</tr>
</tbody>
</table>

Table 4: Voltage: representation in the GRG format and units.

```json
"voltage": {
  "type": "object",
  "required": ["magnitude", "angle"],
  "additionalProperties": true,
  "properties": {
    "magnitude": {"$ref": "/#/values/abstract_value"},
    "angle": {"$ref": "/#/values/abstract_value"}
  }
}
```
In the following example, magnitude and phase angle are defined as variables:

```json
"voltage" : {
    "magnitude" : { "var" : { "lb": 210, "ub": 250.0 } },
    "angle" : { "var" : { "lb": "-Inf", "ub": "Inf" } }
}
}
```

### 1.3 Limits

#### Current Limits

Transmission lines (see section AC Line) and transformers (see section Transformers) have limited current-carrying capacity, described as ranges of current values \([I_{min}, I_{max}]\) that may be sustained for a duration \(d\).

A current limit is represented in GRG format according to Table 5.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>(I_{min})</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>max</td>
<td>(I_{max})</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>duration</td>
<td>(d)</td>
<td>Seconds (sec)</td>
</tr>
</tbody>
</table>

Table 5: Current Limits: representation in the GRG format and units.

A current limit object is expressed as a *table* whose *arguments* are *duration*, *min*, *max*, and *report*. The *report* argument is a boolean field indicating whether the status of the branch should be signaled to the operator (‘on’) or not (‘off’).

```
"current_limits": {
    "type" : "object",
    "required": ["position", "steps"],
    "additionalProperties": true,
    "properties": {
        "arguments": {
            "type" : "array",
            "items": ["duration", "min", "max", "report"]
        },
        "values": {
            "type" : "array",
            "items": [{
                "type": "array",
                "items": [{"$ref": "/values/extended_number"},
                           {"$ref": "/values/extended_number"},
                           {"$ref": "/values/extended_number"},
                           {"enum": ["on", "off"]}],
            },
            "minItems": 4, "maxItems": 4, "additionalItems": "false"
        }
    },
    "minItemes": 1
}
```  

An example of current limits is provided below. The branch can (1) carry up to 563 A indefinitely, or (2) carry between 563 A and 746 A indefinitely, but with a signal to the system operators, or (3) carry current in excess of 746 A for at most 6300 seconds before tripping the branch.
Thermal Limits

Transmission line and transformer limits can also be described in terms of power \([P_{\text{min}}, P_{\text{max}}]\) that may be safely carried for a given duration \(d\). The GRG representation of a thermal limit is provided in Table 6.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>(S_{\text{min}})</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>max</td>
<td>(S_{\text{max}})</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>duration</td>
<td>(d)</td>
<td>Seconds (sec)</td>
</tr>
</tbody>
</table>

Table 6: Thermal Limits: representation in the GRG format and units.

Similar to the current limit, a thermal limit object is expressed as a table whose arguments are: duration, min, max, and report. As with the current limit, report is a boolean indicating whether the status of the branch should be signaled to the operator (‘on’) or not (‘off’).

2 Network Components

Having described the fundamental parameters of electrical devices like transmission lines, transformers, and generators, we now describe these components themselves.
AC Line

An AC line connects network devices in two different points of the network. It has two sides: side 1 is defined as the sending side, and side 2 is defined as the receiving side. The current is positive when flowing from side 1 to side 2. The currents $I_1$ (at side 1) and $I_2$ (at side 2) are given by:

$$I_1 = Y_1 \cdot V_1 + \frac{1}{Z}(V_1 - V_2)$$  \hspace{1cm} (5)$$

$$I_2 = Y_2 \cdot V_2 + \frac{1}{Z}(V_1 - V_2),$$  \hspace{1cm} (6)$$

where $Y_1 = G_1 + jB_1$ and $Y_2 = G_2 + jB_2$ are the admittances at sides 1 and 2, respectively, $Z = R + jX$ is the line impedance, and $V_1$ and $V_2$ are the voltages at the connecting points. Figure 1 illustrates an AC line between nodes 1 and 2 (represented by the black points at the ends).

The power $S_i$ at side $i$ ($i \in \{1, 2\}$) is given by:

$$S_i = \bar{I}_i \cdot V_i$$  \hspace{1cm} (7)$$

Line current/thermal limits are monitored at both ends. For current limits, the absolute value of current magnitude is compared to a sequence of current limits $L_{1} = L_{1,1}, \ldots, L_{1,n}$, and $L_{2} = L_{2,1}, \ldots, L_{2,n}$, where each $L_{ik}$ ($i \in \{1, 2\}, k \in \{1, \ldots, n\}$) is a current limit object, and for each $L_{ik}$, ($k > 1$), $I_{ik}^{\text{min}} = I_{ik-1}^{\text{max}}$. Thermal limits are similar. The absolute value of complex power magnitude is compared to a sequence of thermal limits $L_{1} = L_{1,1}, \ldots, L_{1,n}$, and $L_{2} = L_{2,1}, \ldots, L_{2,n}$, where each $L_{ik}$ ($i \in \{1, 2\}, k \in \{1, \ldots, n\}$) is a thermal limit object, and for each $L_{ik}$, ($k > 1$), $S_{ik}^{\text{min}} = S_{ik}^{\text{max}}$. Finally, the current/thermal limit durations are such that $d_{i1} = \infty$, denoting that $L_{i1}$ is a permanent acceptable limit.

```json
"ac_line": {
  "type": "object",
  "required": ["type", "id", "link_1", "link_2", "shunt_1", "shunt_2", "impedance"],
  "additionalProperties": true,
  "properties": {
    "type": {"enum": ["ac_line"]},
    "id": {"type": "string"},
    "description": {"type": "string"},
    "link_1": {"type": "string"},
    "link_2": {"type": "string"},
    "shunt_1": {"$ref": "/electrical_values/admittance"},
    "shunt_2": {"$ref": "/electrical_values/admittance"},
    "impedance": {"$ref": "/electrical_values/impedance"},
    "current_limits_1": {"$ref": "/limits/current_limits"},
    "current_limits_2": {"$ref": "/limits/current_limits"},
    "thermal_limits_1": {"$ref": "/limits/thermal_limits"},
    "thermal_limits_2": {"$ref": "/limits/thermal_limits"}
  }
}
```
For a component object in the GRG schema, we use `type` and `subtype` (optional) to identify the type of an object (for example an AC line). `id` is a unique identifier for global referencing, and `description` (optional) is further used to describe the component. Fields `link_1` and `link_2` are global identifiers for identifying the voltage points being connected to the network component. For an AC line, `link_1` and `link_2` will be the voltage points at side 1 and side 2 respectively. `shunt_1` and `shunt_2` define the admittances `Y_1` and `Y_2` at sides 1 and 2, respectively. `impedance` defines the impedance `Z` of the line. `current_limits_1` and `current_limits_2` describe collections of current limits associated to sides 1 and 2 of the line. Finally, `thermal_limits_1` and `thermal_limits_2` describe collections of thermal limits associated to sides 1 and 2 of the line. Thermal limits are optional. Starting from GRGv1.6, current limits are also optional. A summary of these AC line components is provided in Table 7.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>link_1</td>
<td>side 1</td>
<td></td>
</tr>
<tr>
<td>link_2</td>
<td>side 2</td>
<td></td>
</tr>
<tr>
<td>impedance → resistance</td>
<td>R</td>
<td>Ohm (Ω)</td>
</tr>
<tr>
<td>impedance → reactance</td>
<td>X</td>
<td>Ohm (Ω)</td>
</tr>
<tr>
<td>shunt_1 → conductance</td>
<td>G_1</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>shunt_1 → susceptance</td>
<td>B_1</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>shunt_2 → conductance</td>
<td>G_2</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>shunt_2 → susceptance</td>
<td>B_2</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>current_limits_1 → values[k][0]</td>
<td>d_{1k}</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>current_limits_1 → values[k][1]</td>
<td>I_{1k}^{\text{min}}</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_1 → values[k][2]</td>
<td>I_{1k}^{\text{max}}</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_2 → values[k][0]</td>
<td>d_{2k}</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>current_limits_2 → values[k][1]</td>
<td>I_{2k}^{\text{min}}</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_2 → values[k][2]</td>
<td>I_{2k}^{\text{max}}</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>thermal_limits_1 → values[k][0]</td>
<td>d_{1k}</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>thermal_limits_1 → values[k][1]</td>
<td>S_{1k}^{\text{min}}</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_1 → values[k][2]</td>
<td>S_{1k}^{\text{max}}</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_2 → values[k][0]</td>
<td>d_{2k}</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>thermal_limits_2 → values[k][1]</td>
<td>S_{2k}^{\text{min}}</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_2 → values[k][2]</td>
<td>S_{2k}^{\text{max}}</td>
<td>MegaWatt (MW)</td>
</tr>
</tbody>
</table>

Table 7: AC line: representation in the GRG format and units.

An example of an AC line in GRG format is provided in Figure 2.

## 2.1 Transformers

### Two Winding Transformer: T model

A two winding transformer connects devices located at two different voltage levels of the network. We denote these voltage levels VL_1 and VL_2, and their associated nominal voltage values `v_1^{\text{nom}}` and `v_2^{\text{nom}}`. Most transformers are equipped with taps on their winding to adjust the voltage transformation or the reactive flow through the transformer. For a specific tap `k`, we denote the voltage magnitude ratio for a transformer to be `r_k`, phase shift to be `\delta_k`, impedance to be `Z_k`, and admittance to be `Y_k`. The current GRG schema supports to two circuit diagram for representing a two winding transformer: the T model (Figure 3) and the PI model (Figure 4). We describe the T model first, then later the PI model.

Without loss of generality, we assume the current is positive when flowing from 1 to 2 (i.e. left to right). The
Example: AC Line

```
"ac_line_11": {
  "type": "ac_line",
  "subtype": "overhead",
  "id": "line_11",
  "link_1": "voltage_id_ALH_5",
  "link_2": "voltage_id_QQF_5",
  "current_limits_1": {
    "arguments": ["duration", "min", "max", "report"],
    "values": ["Inf", 0, 563, "off"],
    ["Inf", 563, 746, "on"],
    [6300, 746, "Inf", "off"]
  },
  "current_limits_2": {
    "arguments": ["duration", "min", "max", "report"],
    "values": ["Inf", 0, 563, "off"]
  },
  "impedance": {"reactance": 6.52, "resistance": 2.39},
  "shunt_1": {"conductance": 0, "susceptance": 2.3e-05},
  "shunt_2": {"conductance": 0, "susceptance": 2.3e-05}
}
```

Figure 2: An example of a GRG AC Line.

![Figure 2](image)

Figure 3: Illustration of the T model for a two winding transformer.

![Figure 3](image)

currents $I_1$ and $I_2$ at sides 1 and 2:

\[ I_1 = \bar{\rho}_k \cdot (Y_k \cdot V_1' + \frac{1}{Z_k} (V_1' - V_2)) \]  \hspace{1cm} (8)

\[ I_2 = \frac{1}{Z_k} (V_1' - V_2), \text{ where} \]  \hspace{1cm} (9)

\[ \rho_k = \left( \frac{V_{1 \text{nom}}}{V_{1 \text{nom}}} \right) \cdot \bar{r}_k \cdot e^{j\delta_k}, V_1' = V_1 \rho_k \]  \hspace{1cm} (10)

$Y_k = G_k + jB_k$ is the admittance of the transformer on tap k, $Z_k = R_k + jX_k$ is the impedance of the transformer on tap k, $V_1$ and $V_2$ are the voltages at the connecting buses in VL1 and VL2, respectively. We use $V_{1 \text{nom}}$ and $V_{2 \text{nom}}$ to denote the nominal voltage magnitudes at sides 1 and 2 of the transformer, respectively. The power $S_i$ at sides i ($i \in \{1, 2\}$) is given by:

\[ S_i = \bar{I}_i \cdot V_i \]  \hspace{1cm} (11)

As with AC lines, transformers have also both current and thermal limits. The maximum absolute value of the current magnitude is monitored on both sides of the branch. Constraints are described through a sequence of current
limits \( L_1 = L_{11}, \ldots, L_{1n} \), and \( L_2 = L_{21}, \ldots, L_{2n} \), where each \( L_{ik} \) \( (i \in \{1, 2\}, k \in \{1, \ldots, n\}) \) is a current limit object, and for each \( L_{ik} \), \( (k > 1) \), \( I_{ik}^{\text{min}} = I_{ik-1}^{\text{max}} \). The absolute value of complex power magnitude is similar and also monitored on both sides of the branch. Constraints are described through a sequence of thermal limits \( L_1 = L_{11}, \ldots, L_{1n} \), and \( L_2 = L_{21}, \ldots, L_{2n} \), where each \( L_{ik} \) \( (i \in \{1, 2\}, k \in \{1, \ldots, n\}) \) is a thermal limit object, and for each \( L_{ik} \), \( (k > 1) \), \( S_{ik}^{m} = S_{ik-1}^{m} \). Finally, the current/thermal limit durations are such that \( d_{ik} = \infty \), denoting that \( L_{ik} \) is a permanent acceptable limit.

**Two Winding Transformer: PI model**

The PI model (Figure 4) are widely used in the literature, and differs from the T model by moving half of the admittance \( Y_k \) to the right hand side. The complex turns ratio \( r_k e^{j\delta_k} \) are also used differently. The PI model represents a step-down transformer scaling down the voltage, while the T model represents a step-up transformer scaling up the voltage.

![Figure 4: Illustration of the PI model (right) for a two winding transformer.](image)

Without loss of generality, we assume the current is positive when flowing from 1 to 2 (i.e. left to right). The currents \( I_1 \) and \( I_2 \) at sides 1 and 2 are:

\[
I_1 = \rho_k \cdot \left( \frac{Y_k}{2} \cdot V_1' + \frac{1}{Z_k} (V_1' - V_2) \right) \tag{12}
\]

\[
I_2 = \frac{1}{Z_k} (V_1' - V_2) - \frac{Y_k}{2} \cdot V_2, \quad \text{where} \tag{13}
\]

\[
\rho_k = \left( \frac{V_2^{\text{nom}}}{V_1^{\text{nom}}} \right) \cdot \frac{1}{r_k} \cdot e^{-j\delta_k}, V_1' = V_1 \rho_k \tag{14}
\]

\( Y_k = G_k + jB_k \) is the admittance of the transformer on tap \( k \) (half of each side), \( Z_k = R_k + jX_k \) is the impedance of the transformer on tap \( k \), \( V_1 \) and \( V_2 \) are the voltages at the connecting buses in \( VL_1 \) and \( VL_2 \), respectively. Similarly, we again use \( V_1^{\text{nom}} \) and \( V_2^{\text{nom}} \) to denote the nominal voltage magnitudes at sides 1 and 2 of the transformer, respectively.

The power \( S_i \) at sides \( i \) \( (i \in \{1, 2\}) \) is again given by:

\[
S_i = I_i \cdot V_i \tag{15}
\]

The thermal and current capacity limits are the same as the T model. Since the T model and the PI model share the same set of components, we use the same JSON format and schema to store and represent both types of transformers.

---

GRG schema: two_winding_transformer

```
"two_winding_transformer": {
  "type": "object",
  "required": ["type", "id", "link_1", "link_2", "tap_changer"],
```

13
In the two winding transformer GRG schema, `type` identify the type of an object (for example a `two_winding_transformer`), and `subtype` identify whether its a T model transformer (Tmodel) or a PI model transformer (PI_model). `id` is a unique identifier for global referencing, and `description` (optional) is further used to describe the component. Fields `link_1` and `link_2` are global identifiers for identifying the voltage points being connected to the network component. For a two winding transformer, `link_1` and `link_2` will be the voltage points at side 1 and side 2.
side 2 respectively. current_limits_1 and current_limits_2 describe collections of current limits associated to sides 1 and 2 of the line. thermal_limits_1 and thermal_limits_2 describe collections of thermal limits associated to sides 1 and 2 of the line. Both current and thermal limits are optional in the schema for transformers. Finally, tap_changer describes the transformer taps. The position field describes the index of the tap step in which the transformer is positioned, and the steps is a table object describing the possible tuple assignments for: 1) the position index, 2) the impedance (impedance), 3) the admittance (shunt), 4) the tap ratio magnitude (tap_ratio), and 5) the tap phase shifts (angle_shift) of the transformer’s tap.

A summary of the two winding transformer components in nominal units is provided in Table 8, and an example is provided in Figure 5.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>link_1</td>
<td>side 1</td>
<td></td>
</tr>
<tr>
<td>link_2</td>
<td>side 2</td>
<td></td>
</tr>
<tr>
<td>current_limits_1→values[k] [0]</td>
<td>d1_k</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>current_limits_1→values[k] [1]</td>
<td>t1_max</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_1→values[k] [2]</td>
<td>t1_min</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_2→values[k] [0]</td>
<td>d2_k</td>
<td>Seconds (sec)</td>
</tr>
<tr>
<td>current_limits_2→values[k] [1]</td>
<td>t2_max</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>current_limits_2→values[k] [2]</td>
<td>t2_min</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>thermal_limits_1→values[k] [0]</td>
<td>s1_k</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_1→values[k] [1]</td>
<td>s1_max</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_1→values[k] [2]</td>
<td>s1_min</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>thermal_limits_2→values[k] [0]</td>
<td>s2_k</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_2→values[k] [1]</td>
<td>s2_max</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_2→values[k] [2]</td>
<td>s2_min</td>
<td></td>
</tr>
<tr>
<td>tap_changer→position</td>
<td>k</td>
<td></td>
</tr>
<tr>
<td>tap_changer→impedance→resistance</td>
<td>R_k</td>
<td>Ohm (Ω)</td>
</tr>
<tr>
<td>tap_changer→impedance→reactance</td>
<td>X_k</td>
<td>Ohm (Ω)</td>
</tr>
<tr>
<td>tap_changer→shunt→conductance</td>
<td>G_k</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>tap_changer→shunt→susceptance</td>
<td>B_k</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>tap_changer→tap_ratio</td>
<td>(\frac{v_m^{nom}}{V_1^{nom}}) r_k (T model)</td>
<td>voltage-ratio</td>
</tr>
<tr>
<td></td>
<td>(\frac{v_m^{nom}}{V_2^{nom}}) r_k (PI model)</td>
<td>voltage-ratio</td>
</tr>
<tr>
<td></td>
<td>\delta_k</td>
<td>degrees</td>
</tr>
</tbody>
</table>

Table 8: Two winding transformer: representation in the GRG format and units.

Three winding Transformer

A three winding transformer connects devices located at three different voltage levels of the network. We denote these voltage levels VL_1, VL_2, and VL_3 and their associated nominal voltage values as v_1^{nom}, v_2^{nom}, and v_3^{nom}, respectively.

We define side 1 as the high voltage side, side 2 as the medium voltage side, and side 3 as the low voltage side. For simplicity, we assume the current flows from side 1 to side 2 and side 3 (i.e. flows from the left to the right in Figure 6).

The current at sides 1, 2, and 3 is given, respectively by:

\[ I_1 = \frac{1}{Z_1} (V_1 - V_m) \]  \hspace{1cm} (16)

\[ I_2 = \rho_2 \frac{1}{Z_2} (V_m - V_2 \rho_2) \]  \hspace{1cm} (17)

\[ I_3 = \rho_3 \frac{1}{Z_3} (V_m - V_3 \rho_3) \]  \hspace{1cm} (18)

where Y_k = G_k + j B_k is the admittance at side 1 of the transformer with tap k; Z_2k = R_2 + j X_2 and Z_3k = R_3 + j X_3 are the impedance values of sides 2 and 3 of the transformer with tap k; V_1, V_2, and V_3 are the voltages at the connecting
Example: Two Winding Transformer

```
"two_winding_transformer_1": {
  "type": "two_winding_transformer",
  "subtype": "T_model",
  "id": "transformer_1",
  "link_1": "voltage_id_9",
  "link_2": "voltage_id_11",
  "current_limits_1": {
    "arguments": ["duration", "min", "max", "report"],
    "values": [
      ["Inf", 0, 1029, "off"],
      [1200, 1029, 1342, "off"],
      [300, 1342, 1790, "off"],
      [60, 1790, "Inf", "off"]
    ]
  },
  "tap_changer": {
    "position": { "var": { "lb": 0, "ub": 0 }},
    "impedance": { "resistance": { "var": { "lb": 0.0, "ub": 0.0 }},
      "reactance": { "var": { "lb": 0.25, "ub": 0.25 }},
    "shunt": { "conductance": { "var": { "lb": 0.0, "ub": 0.0 }},
      "susceptance": { "var": { "lb": 0.0, "ub": 0.0 }},
    "tap_ratio": { "var": { "lb": 0.0, "ub": 11.0 }},
    "angle_shift": { "var": { "lb": 0.0, "ub": 0.0 }},
    "steps": {
      "arguments": ["position","impedance","shunt","tap_ratio","angle_shift"],
      "values": [
        [0, { "resistance": 0.0, "reactance": 0.25167999999999999, "susceptance": 0.0, "conductance": 0.0 }]
      ]
    }
  }
}
```

Figure 5: An example of a GRG Two Winding Transformer.

buses; $V_m$ is the voltage phasor in the star middle point; $\rho_{2k}$ and $\rho_{3k}$ are the complex ratios on sides 2 and 3 of the transformer, defined as:

$$
\rho_{2k} = \left(\frac{V_{2\text{nom}}}{V_{1\text{nom}}}\right)^{r_{2k}} \cdot e^{j\delta_{2k}}
$$

$$
\rho_{3k} = \left(\frac{V_{3\text{nom}}}{V_{1\text{nom}}}\right)^{r_{3k}} \cdot e^{j\delta_{3k}}
$$

(19) 

(20)

corresponding to the $k$-th tap; and $V_{1\text{nom}}$, $V_{2\text{nom}}$, and $V_{3\text{nom}}$ denote the nominal voltage magnitudes at sides 1, 2, and 3 of the transformer. Figure 6 provides an illustration of a three winding transformer.

The power $S_i$ at sides $i$ ($i \in \{1, 2, 3\}$) is given by:

$$
S_i = \bar{I}_i \cdot V_i
$$

(21)
Figure 6: Illustration of a three winding transformer.
In the three winding transformer GRG schema, type and subtype (optional) identify the type of the object (for example a three winding transformer). id is a unique identifier for global referencing, and description is an optional field to describe the component. Fields link_1, link_2, and link_3 are global identifiers for identifying the voltage points being connected to the component. For a three winding transformer, link_1, link_2, and link_3 will be the voltage points at side 1, 2 and 3 respectively. shunt defines the admittance Y and impedance_1 defines the impedance Z_1 of branch 1. current_limits_1, current_limits_2, and current_limits_3 describe collections of current limits associated to sides 1, 2, and 3. thermal_limits_1, thermal_limits_2, and thermal_limits_3 describe collections of thermal limits associated to sides 1, 2, and 3 accordingly. Again similar to two windings transformer, both type of limits are optional. Finally, tap_changer describes the transformer taps. The position field describes the index of the tap step in which the transformer is positioned, and the steps is a table object describing the possible tuple assignments for: 1) the position index, 2) the impedance (impedance_2 & impedance_3), 3) the tap ratio magnitude (tap_ratio_2 & tap_ratio_3), and 4) the tap phase shifts (angle_shift_2 & angle_shift_3) of the transformer’s tap. A summary of three winding transformer components is provided in Table 9, and Figure 7 provides an example.

Switch

A switch is an electrical component that can interrupt the current in a circuit. There are two types of switches: circuit breakers and isolators. Circuit breakers can be switched on or off when they are energized, while isolators (also called disconnectors) can be switched only when not energized. These are series devices with two sides denoted side 1 and side 2. Their operation satisfies the following:

\[ s \cdot V_1 = s \cdot V_2, \]  

(22)

where \( s \) is a binary variable denoting the switch status (1 for open, 0 for closed), and \( V_1 \) and \( V_2 \) are the voltages at sides 1 and 2 of the switch.

Figure 8 illustrates an open isolator (a) and breaker (c), and a closed isolator (b) and breaker (d).
The field `type` identifies the type of the object (switch), `subtype` denotes whether the switch is a breaker or an isolator (disconnector), `description` is an optional field for describing the switch, `id` is a unique identifier, `status` denotes whether the switch is open or closed, and `link_1` and `link_2` identify the connecting voltage points at sides 1 and side 2 of the switch.

Table 9 summarizes switch components, and Figure 9 provides an example.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>link_1</td>
<td>side 1</td>
<td>side 1</td>
</tr>
<tr>
<td>link_2</td>
<td>side 2</td>
<td>side 2</td>
</tr>
<tr>
<td>link_3</td>
<td>side 3</td>
<td>side 3</td>
</tr>
<tr>
<td>impedance_1→resistance</td>
<td></td>
<td>$R_1$ Ohm (Ω)</td>
</tr>
<tr>
<td>impedance_1→reactance</td>
<td></td>
<td>$X_1$ Ohm (Ω)</td>
</tr>
<tr>
<td>shunt→conductance</td>
<td></td>
<td>$G$ Siemens (S)</td>
</tr>
<tr>
<td>shunt→susceptance</td>
<td></td>
<td>$B$ Siemens (S)</td>
</tr>
<tr>
<td>current_limits_1→values[k][0]</td>
<td>$d_{1k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>current_limits_1→values[k][1]</td>
<td>$I_{1k}^\text{min}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>current_limits_1→values[k][2]</td>
<td>$I_{1k}^\text{max}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>current_limits_2→values[k][0]</td>
<td>$d_{2k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>current_limits_2→values[k][1]</td>
<td>$I_{2k}^\text{min}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>current_limits_2→values[k][2]</td>
<td>$I_{2k}^\text{max}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>current_limits_3→values[k][0]</td>
<td>$d_{3k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>current_limits_3→values[k][1]</td>
<td>$I_{3k}^\text{min}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>current_limits_3→values[k][2]</td>
<td>$I_{3k}^\text{max}$ Ampere (A)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_1→values[k][0]</td>
<td>$S_{1k}^\text{min}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_1→values[k][1]</td>
<td>$S_{1k}^\text{max}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_1→values[k][2]</td>
<td>$d_{1k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_2→values[k][0]</td>
<td>$S_{2k}^\text{min}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_2→values[k][1]</td>
<td>$S_{2k}^\text{max}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_2→values[k][2]</td>
<td>$d_{2k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_3→values[k][0]</td>
<td>$S_{3k}^\text{min}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_3→values[k][1]</td>
<td>$S_{3k}^\text{max}$ MegaWatt (MW)</td>
<td></td>
</tr>
<tr>
<td>thermal_limits_3→values[k][2]</td>
<td>$d_{3k}$ Seconds (sec)</td>
<td></td>
</tr>
<tr>
<td>tap_changer→position</td>
<td></td>
<td>$k$</td>
</tr>
<tr>
<td>tap_changer→impedance_2→resistance</td>
<td>$R_{2k}$ Ohm (Ω)</td>
<td></td>
</tr>
<tr>
<td>tap_changer→impedance_2→reactance</td>
<td>$X_{2k}$ Ohm (Ω)</td>
<td></td>
</tr>
<tr>
<td>tap_changer→impedance_3→resistance</td>
<td>$R_{3k}$ Ohm (Ω)</td>
<td></td>
</tr>
<tr>
<td>tap_changer→impedance_3→reactance</td>
<td>$X_{3k}$ Ohm (Ω)</td>
<td></td>
</tr>
<tr>
<td>tap_changer→tap_ratio_2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tap_changer→tap_ratio_3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tap_changer→angle_shift_2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tap_changer→angle_shift_3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9: Three winding transformer: representation in the GRG format and units.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>s</td>
<td>boolean</td>
</tr>
</tbody>
</table>

Table 10: Switch representation in GRG format and units.
Example: Three Winding Transformer

```
"three_winding_transformer_tr1" : {
  "type" : "three_winding_transformer",
  "id" : "transformer_1",
  "link_1" : "voltage_id_2",
  "link_2" : "voltage_id_3",
  "link_3" : "voltage_id_6",
  "impedance_1" : {"reactance" : 60.1, "resistance" : 1.512},
  "shunt" : {"conductance" : 0, "susceptance" : 0},
  "tap_changer": {
    "position": { "var": { "lb": 0, "ub": 0}},
    "impedance_2": { "resistance": { "var": { "lb": 0.0,"ub": 0.0}}},
    "impedance_3": { "resistance": { "var": { "lb": 0.25,"ub": 0.25}}},
    "tap_ratio_2": {"var": {"lb": 0.0,"ub": 11.0 }},
    "impedance_3": {"var": {"lb": 0.0,"ub": 9.0 }},
    "angle_shift_2": {"var": {"lb": 0.0,"ub": 0.0 }},
    "angle_shift_3": {"var": {"lb": 0.0,"ub": 0.0 }},
    "steps": {
      "arguments": ["position","impedance_2","tap_ratio_2",
                    "angle_shift_2","impedance_3","tap_ratio_3", "angle_shift_3" ],
      "values": [ 0,
                  { "resistance": 0.0,"reactance": 0.25 },
                  11.0,
                  0.0,
                  { "resistance": 0.0,"reactance": 0.21 },
                  9.0,
                  0.0,
                  ]
    }
  }
}
```

Figure 7: An example of a GRG Three Winding Transformer.

![Diagram](image)

Figure 8: Illustration of an open (a) and close (b) isolator, and an open (c) and close (d) breaker.

**Bus**

A bus is a set of equipment connected together. It could be a configured object or the result of a computation, depending of the context.

```
"bus": {
  "type" : "object",
  "required": ["type", "id", "link", "voltage"],
```
Example: Switch

```
"switch_F" : {
    "type" : "switch",
    "subtype" : "breaker",
    "id" : "sw_1",
    "link_1" : "voltage_id_C_0",
    "link_2" : "voltage_id_C_3",
    "status" : {"var" : ["off", "on"]}
}
```

Figure 9: An example of a GRG switch.

```
"additionalProperties": true,
"properties": {
    "type" : {"enum": ["bus"]},
    "subtype" : {"type": "string"},
    "description" : {"type": "string"},
    "id" : {"type": "string"},
    "link" : {"type": "string"},
    "voltage" : {"$ref": "/#values/electrical_values/voltage"},
    "name" : {"type": "string"},
}
```

Type identifies the type of the object. Depending on the network topology adopted, a bus can be represented as a busbar (node-breaker topology), logical bus (bus-breaker topology), or simply bus (bus-branch topology). These information can be stored in the subtype optional field. Description is an optional description field, and id is a unique identifier. The field link identifies the voltage point at which the bus is connected. Finally, voltage refers to the bus voltage magnitude \( v \) and phase angle \( \theta \), and name denotes the bus name.

Table 11 tabulates the bus components, while Figure 10 contains an example of a bus in GRG format.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltage→magnitude v</td>
<td>kiloVolt (kV)</td>
<td></td>
</tr>
<tr>
<td>voltage→angle→lb Θ</td>
<td>Degrees</td>
<td></td>
</tr>
<tr>
<td>voltage→angle→ub Θ</td>
<td>Degrees</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Bus representation in GRG format and units.

Example: Busbar

```
"bus_Q" : {
    "type" : "bus",
    "id" : "bus_6",
    "link" : "voltage_id_P_0",
    "voltage" : {
        "angle" : {"var" : {"lb" : -30.0, "ub" : 30.0}},
        "magnitude" : {"var" : {"lb" : 0, "ub" : 500}}
    }
}
```

Figure 10: An example of a GRG busbar.

Shunt

A shunt capacitor or reactor is defined as an admittance:

\[
I = -Y \cdot V,
\]

where \( Y = G + jB \) is the admittance, and \( V \) is the voltages at the connecting point. Figure 11 illustrates a shunt.
In the above schema, `type` identifies the type of the object (i.e., a shunt), `id` is a unique identifier, and `description` is an optional description field. The field `link` identifies the connecting voltage point. Finally, `shunt` defines the admittance $Y$.

Table 12 summarizes shunt components, while Figure 12 provides an example.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>shunt $\rightarrow$ conductance</td>
<td>$G$</td>
<td>Siemens (S)</td>
</tr>
<tr>
<td>shunt $\rightarrow$ susceptance</td>
<td>$B$</td>
<td>Siemens (S)</td>
</tr>
</tbody>
</table>

Table 12: Shunt representation in the GRG format and units.

Example: Shunt

```
"shunt_IO": {
  "id" : "sh_10",
  "type" : "shunt",
  "subtype": "inductor",
  "link" : "voltage_id_HS_9",
  "shunt": {"conductance" : 0.0, "susceptance" : -0.16}
}
```

Figure 12: An example of a GRG shunt.

**Load**

A Load consumes active power $P$ and reactive power $Q$ at its connection point, as illustrated in Figure 13.

```
"load": {
  "type": "object",
  "required": ["type", "id", "link", "demand"],
  "additionalProperties": true,
```
In the load GRG schema, `type` identifies the type of the object (i.e., a load), `subtype` identifies the type of load (for example `withdrawal` to indicate the load will only withdraw power). The attribute `id` is a unique identifier for the load component, and `description` is an optional description field. The field `link` identifies the connecting voltage point in the network. Finally, `demand` defines the power $S$ consumed by the load, and can be set to a variable to be assigned later.

Table 13 summarizes load components, while Figure 14 provides an example.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>demand→active</td>
<td>$P$</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>demand→reactive</td>
<td>$Q$</td>
<td>MegaVolt-Ampere Reactive (MVAR)</td>
</tr>
</tbody>
</table>

Table 13: Load representation in GRG format and units.

---

Figure 14: An example of a GRG load.

**Generator**

A generator produces active power ($P$) and reactive power ($Q$), and supports voltage ($V$). Its output is limited according to a PQ-curve, which specifies minimum and maximum reactive power values for every active power value. More formally, a PQ-curve is a sequence of tuples: $(P_k, Q_{min}^k, Q_{max}^k)_{k=1}^n$ (with $n > 0$) such that for each $k < n$, $P_k < P_{k+1}$, and $Q_{min}^k \leq Q_{max}^k$.

Figure 15 illustrates a generator (a) and feasible PQ region (shaded gray area) of its PQ-curve.

---

Figure 15: An example of a PQ-curve.
Figure 15: Illustration of a generator (a) and its PQ-curve (b).

In the generator GRG schema, type identifies the type of the object (i.e., a generator), subtype identifies the type of generator, id is a unique identifier, and description is an optional description field. link identifies the connecting voltage point in the network. The output field defines the power $S$ produced by the generator. Finally, the PQ-curve is a table of active and reactive bounds, defining the feasible operating region.

Table 14 summarizes generator components, and Figure 16 provides an example.

**Synchronous Condenser**

A synchronous condenser is a spinning machine that can compensate lagging current by either generating or absorbing reactive power.
Table 14: Generator: representation in the GRG format and units.

Example: Generator

```json
"generator_AEC" : {
    "type" : "generator",
    "subtype": "solar",
    "id" : "gen_4",
    "link" : "voltage_id_ACM_5",
    "PQ_curve" : {
        "arguments": ["active", "reactive_lb", "reactive_ub"],
        "values" : [[0.0, 0.0, 0.0],
                     20.7, 0.0, 0.0]]
    }
}
```

Figure 16: An example of a GRG generator.

In the synchronous condenser GRG schema, type and subtype identifies the type of the object, the attribute id is a unique identifier, and description is an optional description field. link identifies the connecting voltage point in the network. output defines the reactive power (reactive) produced or absorbed by the condenser.

Table 15 summarizes synchronous condenser components, and Figure 17 provides an example.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>symbol</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>output → active</td>
<td>$P$</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>output → reactive</td>
<td>$Q$</td>
<td>MegaVolt-Ampere Reactive (MVAR)</td>
</tr>
<tr>
<td>PQ_curve → values[k][0]</td>
<td>$P_k$</td>
<td>MegaWatt (MW)</td>
</tr>
<tr>
<td>PQ_curve → values[k][1]</td>
<td>$Q_{E \text{min}}$</td>
<td>MegaVolt-Ampere Reactive (MVAR)</td>
</tr>
<tr>
<td>PQ_curve → values[k][2]</td>
<td>$Q_{E \text{max}}$</td>
<td>MegaVolt-Ampere Reactive (MVAR)</td>
</tr>
</tbody>
</table>

Table 15: Synchronous condenser representation in GRG format and units.
Example: Synchronous Condenser

```
"sync_cond_3" : {
    "type" : "synchronous_condenser",
    "id" : "sync_3",
    "link" : "voltage_id_11",
    "output" : {
        "reactive" : {"var" : {"lb" : 0, "ub" : 14.53}}
    }
}
```

Figure 17: An example of a GRG synchronous condenser.

3 Assignment and Mappings

An *assignment* is a JSON object which contains a list of value assignments for some network component. Assignments may be used to define a particular instance of a network, for example, by assigning values to a set of variables. The assignment schema is as follows.

```
GRG schema: assignment

"assignment": {
    "type" : "object",
    "patternProperties": {
        ".*": {
            "oneOf": [
                {"type": "object"},
                {"$ref": "/#/values/basic_values/abstract_value"},
                {"$ref": "/#/values/basic_values/status"}]
        }
    }
}
```

The JSON reference refers to any of the components described in section 2, and the `patternProperties` specifies that zero or more component assignments can be specified.

We say that a network component field is *assigned* if its value is specified as an object in an assignment. An assignment example is shown in Figure 18.

```
Example: Assignment

"assignment": {
    "switch_773/status": {"on"},
    "bus_6/voltage": {
        "angle": 4.23,
        "magnitude": 63.12
    }
}
```

Figure 18: An example of a GRG assignment.

A *mapping* is a set of assignments. It can be used to to define a particular instantiation of a network, or a desired network state, e.g., target values. Its schema is as follows.

```
GRG schema: mappings

"mappings": {
    "type": "object",
    "patternProperties": {
        ".*": {"$ref": "/#/network/network_assignments"}
    }
}
```

The `patternProperties` field specifies that zero or more set of component assignments can be specified. For each of these sets, an arbitrary number of component assignments can be specified. Figure 19 contains an example.
Example: Mappings

```json
"mappings": {
  "starting_points": {
    "bus_6/voltage": {
      "magnitude": 132.22,
      "angle": -10.15
    },
    "ld_7/demand": {
      "active": 5.80,
      "reactive": 2.0
    },
    "sync_3/output": {
      "active": 0.0,
      "reactive": 11.87
    },
    "transformer_1/tap_changer/position": 0
  }
}
```

Figure 19: An example of GRG mappings.

Similar to mappings, an operation constraints block is a set of assignments. It is reserved for posting operational constraints to restrict the range of values allowed to be assigned to variables/states in the network. Its schema is as follows.

```json
"operation_constraints": {
  "type": "object",
  "patternProperties": {
    "/#values/basic_values/grg_pointer": {
      "$ref": "#/network/network_assignments"
    }
  }
}
```

The patternProperties field again specifies that zero or more set of component assignments can be specified. Figure 20 contains an example on how we describe the angle difference constraints on AC lines and two winding transformers, by explicitly restricting the angle difference value has to be within the range of [-30,30] (i.e. 30 degree angle difference constraints).

```json
"operation_constraints": {
  "transformer_6/angle_difference": {
    "var": {
      "lb": -30,
      "ub": 30
    }
  },
  "line_4/angle_difference": {
    "var": {
      "lb": -30,
      "ub": 30
    }
  }
}
```

Figure 20: An example of GRG operation constraints.
4 Networks

A GRG network is a collection of network components, along with a list of component assignments. When a network component is assigned, its value in the assignments block of the GRG document supersedes its value in the network block. The GRG schema for a network is shown below.

```
"network": {
  "type": "object",
  "required": ["id", "type", "subtype", "per_unit", "components"],
  "properties": {
    "id": {"type": "string"},
    "type": {"enum": ["network"]},
    "subtype": {"enum": ["node_breaker", "bus_breaker", "bus_branch"]},
    "per_unit": {"type": "boolean"},
    "description": {"type": "string"},
    "components": {"$ref": "/#/network/network_components"},
    "assignments": {"$ref": "/#/network/network_assignments"},
  },
  "additionalProperties": true
}
```

The attribute type identifies the type of the object (i.e., a network). subtype indicates topology type, which defines the level of detail in descriptions of connections between components; description is an optional description field, and per unit indicates whether network component values are expressed in per unit or nominal value.

A network is organized as a set of substations connected via transmission lines. Together these comprise the network_components block, as shown in the following schema fragment:

```
"network_components": {
  "patternProperties": {
    ".*": {
      "oneOf": [
        {"$ref": "/#/network_components/substation"},
        {"$ref": "/#/network_components/ac_line"}
      ]
    }
  }
}
```

Substation

A substation is a collection of equipment located at a the same physical site and belonging to one Transmission System Operator (TSO). It is composed of several voltage levels and transformers. Its schema is as follows.

```
"substation": {
  "type": "object",
  "required": ["type", "id", "substation_components"],
  "properties": {
    "type": {"enum": ["substation"]},
    "subtype": {"type": "string"},
    "description": {"type": "string"},
    "id": {"type": "string"},
    "country": {"type": "string"},
    "TSO": {"type": "string"},
    "substation_components": {
      "patternProperties": {
        ".*": {
          "oneOf": [
            {"$ref": "/#/network_components/voltage_level"},
            {"$ref": "/#/network_components/two_winding_transformer"}
          ]
        }
      }
    }
  }
}
```
The attribute type identifies the type of the object (i.e., a substation). The attributes country and TSO indicate the country in which the substation is located, along with its Transmission System Operator. Substation components are listed in the creatively-named substation_components object.

**Voltage Level**

A voltage level is a collection of equipment located in the same substation at the same nominal voltage value. Its schema is as follows.

```json
"voltage_level": {
  "type": "object",
  "required": ["type", "id", "voltage", "voltage_points"],
  "properties": {
    "type": {"enum": ["voltage_level"]},
    "subtype": {"type": "string"},
    "description": {"type": "string"},
    "id": {"type": "string"},
    "voltage_points": { "type": "array", "items": { "type": "string" }, "minItems": 1 },
    "voltage": {
      "type": "object",
      "required": ["nominal_value", "upper_limit", "lower_limit"],
      "additionalProperties": true,
      "properties": {
        "nominal_value": {"type": "number"},
        "upper_limit": {"type": "number"},
        "lower_limit": {"type": "number"}
      }
    },
    "voltage_level_components": {
      "patternProperties": { ". *": {
        "oneOf": [
          {"$ref": "/network_components/bus"},
          {"$ref": "/network_components/shunt"},
          {"$ref": "/network_components/generator"},
          {"$ref": "/network_components/synchronous_condenser"},
          {"$ref": "/network_components/load"},
          {"$ref": "/network_components/switch"}
        ]
      }
    }
  }
}
```

The attribute type identifies the type of the object (i.e., a voltage_level). Voltage contains the nominal voltage $V_{nom}$ along with voltage limits $[V_L, V_U]$ for all components contained in the voltage level. Since GRGv1.5, we unify all network components to be connected by voltage points, and voltage points are assume to be global and unique. To allow ease of parsing, we require all voltage levels to explicitly list all its voltage point labels in voltage_points starting from GRG v1.6. The voltage level components are listed in voltage_components.

Table 16 describes the voltage level element.

### 5 Market

A market GRG block attaches operational costs to a network component. A cost JSON block represents a cost model for a component.
<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltage→nominal_value</td>
<td>$V_{nom}$</td>
<td>KiloVolt (KV)</td>
</tr>
<tr>
<td>voltage→lower_limit</td>
<td>$V_L$</td>
<td>KiloVolt (KV)</td>
</tr>
<tr>
<td>voltage→upper_limit</td>
<td>$V_U$</td>
<td>KiloVolt (KV)</td>
</tr>
</tbody>
</table>

Table 16: Voltage Level description.

Each item in the costs block is named with a pointer to its associated network element. The arguments are pointers to network elements, and coefficient items define the associated cost polynomial. The $i$-th row of a coefficients array identifies the coefficients, $a_{n-1},...,a_0$, associated with the $i$-th $x_i$ argument in the arguments array, generating the polynomial:

$$a_{n-1}x_i^{n-1} + \ldots + a_1x_i + a_0$$

Since the cost coefficients are dependent on the units of the arguments array, they will need to be re-computed if the units of the arguments array changed (e.g. during per-unit operations).

6 Units

A units object details the units adopted for each physical quantity. These may differ from unit associated to the description of various components in the previous sections. The JSON schema of units is defined below.
7 GRG Document

A GRG document is defined as a collection of JSON blocks, and includes a network block, a mapping block, a market block, and a units block. Its schema is as follows.

```
GRG schema: GRG document
{
    "type": "object",
    "required": ["grid_version", "network", "units"],
    "additionalProperties": true,
    "properties": {
        "grid_version": {"type": "string"},
        "description": {"type": "string"},
        "network": {"$ref": "#/network"},
        "mappings": {"$ref": "#/mappings"},
        "market": {"$ref": "#/market"},
        "units": {"$ref": "#/units"},
        "operation_constraints": {"$ref": "#/operation_constraints"}
    }
}
```

8 Network Topology

A network may be represented in one of three different topologies, from finer to coarser level of detail: node-breaker, bus-breaker, or bus-branch. Table 17 summarizes these representations, indicating which ones capture the voltage level topology (i.e., whether all the elements and their connections are physical ones) or not (i.e., merely logical connections); which switch types are captured, and the representation of buses. We will now discuss each representation in detail.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Node-breaker</th>
<th>Bus-breaker</th>
<th>Bus-branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breakers</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Disconnectors</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Bus Type</td>
<td>busbar</td>
<td>logical bus</td>
<td>bus</td>
</tr>
</tbody>
</table>

Table 17: Network Topologies.

8.1 Node-Breaker Topology

A node-breaker topology contains the highest level of detail for a network. All components are physical elements, including busbar sections, breakers, and disconnectors. In this topology, a voltage level is a collection of network components and connection nodes. Nodes are logical connection points labeled with values in $\mathbb{N} \cup \{0\}$. Each component is directly connected to one node (if it is a bus, shunt, generator, or load), or to two nodes (if it is an AC
Algorithm 1: Node-Breaker to Bus-Breaker($\mathcal{N}$)

1. foreach voltage level $\mathcal{V} \in \mathcal{N}$ do
2. \quad Transform-VoltageLevel($\mathcal{V}$); \\
3. foreach line and transformer $l \in \mathcal{L}_N \cup \mathcal{T}_N$ do
4. \quad if $\exists$ paths through close switches from $l$ to some $b_1 \in \mathcal{B}_{\mathcal{V}_1}$ and $b_2 \in \mathcal{B}_{\mathcal{V}_2}$ then
5. \quad \quad Remove $l$ from $\mathcal{V}$;

line, transformer, or switch). The connection of a network component to a node is described in its link attribute (or link_1 and link_2 attributes for a connector). The set of nodes in a voltage level is described implicitly: It is the union of link values of all network components in that voltage level. Figure 21 (a) shows a voltage level in the node-breaker topology where breakers are illustrated as white squares, and disconnectors as pairs of white circles.

![Figure 21: Illustration of a voltage level in node-breaker (a), bus-breaker (b), and bus-branch (c) topologies.](image)

8.2 Bus-Breaker Topology

A network in bus-breaker form also represents a voltage level as a collection of components connected through nodes. However, the bus-breaker representation contains no disconnectors. Algorithm 1 describes the procedure for constructing a bus-breaker topology from a network in node-breaker form. The algorithm accepts a node-breaker network $\mathcal{N}$ as input. Lines (1–2) call the procedure Transform-VoltageLevel for each voltage level $\mathcal{V}$ in the network $\mathcal{N}$. The result of this operation is to (1) merge busbars of the voltage level which are connected by closed disconnectors, (2) remove all disconnectors from the voltage level, and (3) remove components that have no path to a logical bus. Lines (6–12) show greater detail for the removal process of a set of disconnectors $\mathcal{D}_\mathcal{V}$ of the voltage level $\mathcal{V}$. A disconnector is removed from the network by linking all voltage level components attached to its node on side 2 (link_2) to its node on side 1 (link_1) (lines 9–11). This operation may result in multiple busbars connected to the same node. For instance, busbars BBS1 and BBS2 will be connected to node 0 when disconnector DI1 is removed. The effect of the loop in lines 13–15 is to merge such busbar sets into single logical buses. This is done by preserving one busbar of the set $B$, changing its type to a logical bus, and removing all other busbars in $B$.

The effect of executing lines 16–18 is to remove all components of the voltage level for which there is no path through closed switches or non-assigned breakers to a logical bus of the voltage level. For instance, in Figure 21 (a), the breakers BR1, BR4, B5, the generator GN1, and the load LD1 have no closed path to a voltage level bus, since disconnector DI2 is open. Thus, these components are removed from the voltage level. The result of this operation is illustrated in Figure 21 (b).

Finally, in lines (3–5) of Algorithm 1, all lines and transformers (connectors) of the network are processed. As was done for voltage levels, a connector is removed if it has no closed path to logical buses on both sides.

Figure 21 (b) illustrates an example voltage level in the bus-breaker topology obtained from the node-breaker topology of Figure 21 (a).
Procedure Transform-VoltageLevel(V)

6 \textbf{foreach} disconnectors \(d \in \mathcal{D}_V\) \textbf{do}
7 \hspace{1em} \(n_1 \leftarrow d[\text{link}_1] ;\)
8 \hspace{1em} \(n_2 \leftarrow d[\text{link}_2] ;\)
9 \hspace{1em} \textbf{foreach} component \(c \in V\) s.t. \(c\) is connected to \(n_2\) \textbf{do}
10 \hspace{2em} Let \(\text{link}\) be \(\text{link}_i\) if \(c\) is a switch, line, or transformer, and \(\text{link}_i \neq n_2;\)
11 \hspace{2em} \(c[\text{link}] \leftarrow n_1 ;\)
12 \hspace{1em} Remove \(d\) from \(V ;\)
13 \hspace{1em} \textbf{foreach} busbar \(b \in \mathcal{B}_V\) \textbf{do}
14 \hspace{2em} Let \(B = \{b' \in \mathcal{B}_V : b'[\text{link}] = b[\text{link}]\} ;\)
15 \hspace{2em} Remove all \(b' \in \mathcal{B} \setminus \{b\} ;\)
16 \hspace{1em} \textbf{foreach} component \(c \in V\) \textbf{do}
17 \hspace{2em} if \(\exists\) path through close switches from \(c\) to some \(b \in \mathcal{B}_V\) then
18 \hspace{3em} Remove \(c\) from \(V ;\)

8.3 Bus-Branch Topology

A network in bus-branch form differs from one in a node- or bus-breaker form in that switches are not represented, and all network components are directly connected to a bus. Thus, the bus-branch topology has no concept of connection nodes.

A bus-branch topology can be obtained from a node- or a bus-breaker topology by applying a variation of Algorithm 1. Rather than processing disconnectors (line 6), all switches of the voltage level are processed. Finally, after the algorithm is executed, the \(\text{link}\) attribute of each generator, load, and shunt, and the \(\text{link}_1\) and \(\text{link}_2\) attributes of lines and transformers are updated to take the ID of the bus to which they are connected.

Figure 21 (c) illustrates an example voltage level in the bus-branch topology, obtained from the bus-breaker topology of Figure 21 (b).

9 Per-Unit Transformations

This section describes all per-unit transformations for network components.

AC Line

The per-unit transformation of an AC line (see also Figure 1) is shown in Table 18. The nominal current magnitude \(I_{\text{nom}}\) of the AC line is defined as:

\[
I_{\text{nom}} = \frac{S_{\text{nom}}}{V_{\text{nom}}} ,
\]

where \(V_{\text{nom}}\) is voltage magnitude on either side of the line, and \(S_{\text{nom}}\) is the nominal power magnitude of the network. The nominal impedance of the AC line is defined as:

\[
Z_{\text{nom}} = \frac{(V_{\text{nom}})^2}{S_{\text{nom}}}
\]
<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>impedance→resistance</td>
<td>$R$</td>
<td>$\frac{1}{Z_{nom}} \cdot R$</td>
</tr>
<tr>
<td>impedance→reactance</td>
<td>$X$</td>
<td>$\frac{1}{Z_{nom}} \cdot X$</td>
</tr>
<tr>
<td>shunt₁→conductance</td>
<td>$G_1$</td>
<td>$Z_{nom} \cdot G_1$</td>
</tr>
<tr>
<td>shunt₁→susceptance</td>
<td>$B_1$</td>
<td>$Z_{nom} \cdot B_1$</td>
</tr>
<tr>
<td>shunt₂→conductance</td>
<td>$G_2$</td>
<td>$Z_{nom} \cdot G_2$</td>
</tr>
<tr>
<td>shunt₂→susceptance</td>
<td>$B_2$</td>
<td>$Z_{nom} \cdot B_2$</td>
</tr>
<tr>
<td>current_limits₁→values[k][1]</td>
<td>$S_{1\text{min}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{I_{1\text{min}}}$</td>
</tr>
<tr>
<td>current_limits₁→values[k][2]</td>
<td>$S_{1\text{max}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{I_{1\text{max}}}$</td>
</tr>
<tr>
<td>current_limits₂→values[k][1]</td>
<td>$S_{2\text{min}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{I_{2\text{min}}}$</td>
</tr>
<tr>
<td>current_limits₂→values[k][2]</td>
<td>$S_{2\text{max}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{I_{2\text{max}}}$</td>
</tr>
<tr>
<td>thermal_limits₁→values[k][1]</td>
<td>$S_{1\text{min}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{S_{1\text{min}}}$</td>
</tr>
<tr>
<td>thermal_limits₁→values[k][2]</td>
<td>$S_{1\text{max}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{S_{1\text{max}}}$</td>
</tr>
<tr>
<td>thermal_limits₂→values[k][1]</td>
<td>$S_{2\text{min}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{S_{2\text{min}}}$</td>
</tr>
<tr>
<td>thermal_limits₂→values[k][2]</td>
<td>$S_{2\text{max}}$</td>
<td>$\frac{1}{S_{\text{nom}}}, \frac{1}{S_{2\text{max}}}$</td>
</tr>
</tbody>
</table>

Table 18: AC line per-unit transformations.

**Two winding transformer**

For a **two winding transformer** (see Figure 3), the per-unit transformation is defined in Table 19. The nominal current magnitudes $I_{1\text{nom}}^1$ on side 1 and $I_{2\text{nom}}^2$ on side 2 are defined as:

$$I_{1\text{nom}}^1 = \frac{S_{\text{nom}}}{V_{1\text{nom}}^1} \quad (25)$$

$$I_{2\text{nom}}^2 = \frac{S_{\text{nom}}}{V_{2\text{nom}}^2} \quad (26)$$

where $V_{1\text{nom}}^1$ and $V_{2\text{nom}}^2$ are the voltage magnitudes on sides 1 and 2 of the transformer. The nominal impedance magnitude on side 2 (low voltage side) of the transformer is defined as:

$$Z_{nom}^2 = \frac{(V_{2\text{nom}}^2)^2}{S_{\text{nom}}} \quad (27)$$

**Three winding transformer**

For a **three winding transformer** (see Figure 6), the per-unit transformation is defined in Table 20. The nominal impedance magnitude on side 1 (high voltage side) of the transformer is defined as:

$$Z_{nom}^1 = \frac{(V_{1\text{nom}}^1)^2}{S_{\text{nom}}} \quad (28)$$

where $V_{1\text{nom}}^1$, $V_{2\text{nom}}^2$, and $V_{3\text{nom}}^3$ are voltage magnitudes at sides 1, 2, and 3 of the transformer.

**Bus**

The per-unit transformation for a bus is defined in Table 21, where $V_{\text{nom}}$ is voltage magnitude at the voltage level of the bus. If the network is in flat representation, then the voltage component of the bus is transformed in per-unit according to the description provided in the last three rows of Table 21.

**Shunt**

For a **shunt** (see Figure 11), the per-unit transformation is defined in Table 22. The nominal impedance of the AC line is defined as:

$$Z_{nom} = \frac{(V_{\text{nom}})^2}{S_{\text{nom}}} \quad (29)$$
Table 19: Two winding transformer per-unit transformations.

Load
For a load, the per-unit transformation is defined in Table 23.

Generator
For a generator, the per-unit transformation is defined in Table 24.

Synchronous Condenser
For a synchronous condenser, the per-unit transformation is defined in Table 25.

Voltage Level
Each voltage component of a voltage level is transformed according to Table 26.
<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>impedance_1→resistance</td>
<td>$R_1$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot R$</td>
</tr>
<tr>
<td>impedance_1→reactance</td>
<td>$X_1$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot X$</td>
</tr>
<tr>
<td>shunt→conductance</td>
<td>$G$</td>
<td>$Z_{\text{sym}} \cdot G$</td>
</tr>
<tr>
<td>shunt→susceptance</td>
<td>$B$</td>
<td>$Z_{\text{sym}} \cdot B$</td>
</tr>
<tr>
<td>tap_changer→steps→impedance_2→resistance</td>
<td>$R_k$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot R_k$</td>
</tr>
<tr>
<td>tap_changer→steps→impedance_2→reactance</td>
<td>$X_k$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot X_k$</td>
</tr>
<tr>
<td>tap_changer→steps→impedance_3→resistance</td>
<td>$R_k^3$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot R_k^3$</td>
</tr>
<tr>
<td>tap_changer→steps→impedance_3→reactance</td>
<td>$X_k^3$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot X_k^3$</td>
</tr>
<tr>
<td>tap_changer→steps→tap_ratio_2</td>
<td>$r_2 k$</td>
<td>$r_2 k$</td>
</tr>
<tr>
<td>tap_changer→steps→tap_ratio_3</td>
<td>$r_3 k$</td>
<td>$r_3 k$</td>
</tr>
<tr>
<td>tap_changer→steps→angle_shift_2</td>
<td>$\delta_2^k$</td>
<td>$\delta_2^k$</td>
</tr>
<tr>
<td>tap_changer→steps→angle_shift_3</td>
<td>$\delta_3^k$</td>
<td>$\delta_3^k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][1]→impedance_2→resistance</td>
<td>$R_k$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot R_k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][1]→impedance_2→reactance</td>
<td>$X_k$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot X_k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][2]→impedance_3→resistance</td>
<td>$R_k^3$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot R_k^3$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][2]→impedance_3→reactance</td>
<td>$X_k^3$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot X_k^3$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][3]→tap_ratio_2</td>
<td>$r_2 k$</td>
<td>$r_2 k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][4]→angle_shift_2</td>
<td>$\delta_2^k$</td>
<td>$\delta_2^k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][5]→tap_ratio_3</td>
<td>$r_3 k$</td>
<td>$r_3 k$</td>
</tr>
<tr>
<td>tap_changer→steps→values[k][6]→angle_shift_3</td>
<td>$\delta_3^k$</td>
<td>$\delta_3^k$</td>
</tr>
<tr>
<td>current_limits_1→values[k][1]</td>
<td>$I_{\text{min}}^1$</td>
<td>$I_{\text{min}}^1$</td>
</tr>
<tr>
<td>current_limits_1→values[k][2]</td>
<td>$I_{\text{max}}^1$</td>
<td>$I_{\text{max}}^1$</td>
</tr>
<tr>
<td>current_limits_2→values[k][1]</td>
<td>$I_{\text{min}}^2$</td>
<td>$I_{\text{min}}^2$</td>
</tr>
<tr>
<td>current_limits_2→values[k][2]</td>
<td>$I_{\text{max}}^2$</td>
<td>$I_{\text{max}}^2$</td>
</tr>
<tr>
<td>current_limits_3→values[k][1]</td>
<td>$I_{\text{min}}^3$</td>
<td>$I_{\text{min}}^3$</td>
</tr>
<tr>
<td>current_limits_3→values[k][2]</td>
<td>$I_{\text{max}}^3$</td>
<td>$I_{\text{max}}^3$</td>
</tr>
<tr>
<td>thermal_limits_1→values[k][1]</td>
<td>$S_{\text{min}}^1$</td>
<td>$S_{\text{min}}^1$</td>
</tr>
<tr>
<td>thermal_limits_1→values[k][2]</td>
<td>$S_{\text{max}}^1$</td>
<td>$S_{\text{max}}^1$</td>
</tr>
<tr>
<td>thermal_limits_2→values[k][1]</td>
<td>$S_{\text{min}}^2$</td>
<td>$S_{\text{min}}^2$</td>
</tr>
<tr>
<td>thermal_limits_2→values[k][2]</td>
<td>$S_{\text{max}}^2$</td>
<td>$S_{\text{max}}^2$</td>
</tr>
<tr>
<td>thermal_limits_3→values[k][1]</td>
<td>$S_{\text{min}}^3$</td>
<td>$S_{\text{min}}^3$</td>
</tr>
<tr>
<td>thermal_limits_3→values[k][2]</td>
<td>$S_{\text{max}}^3$</td>
<td>$S_{\text{max}}^3$</td>
</tr>
</tbody>
</table>

Table 20: Three winding transformer per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltage→magnitude</td>
<td>$v$</td>
<td>$\frac{1}{V_{\text{sym}}} \cdot v$</td>
</tr>
<tr>
<td>voltage→angle→lb</td>
<td>$\theta^l$</td>
<td>$\frac{1}{180} \cdot \theta^l$</td>
</tr>
<tr>
<td>voltage→angle→ub</td>
<td>$\theta^u$</td>
<td>$\frac{1}{180} \cdot \theta^u$</td>
</tr>
<tr>
<td>voltage→nominal_value</td>
<td>$V_{\text{nom}}$</td>
<td>$\frac{1}{V_{\text{sym}}} \cdot V_{\text{nom}}$</td>
</tr>
<tr>
<td>voltage→lower_limit</td>
<td>$V_L$</td>
<td>$\frac{1}{V_{\text{sym}}} \cdot V_L$</td>
</tr>
<tr>
<td>voltage→upper_limit</td>
<td>$V_U$</td>
<td>$\frac{1}{V_{\text{sym}}} \cdot V_U$</td>
</tr>
</tbody>
</table>

Table 21: Voltage component per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>shunt→conductance</td>
<td>$G$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot G$</td>
</tr>
<tr>
<td>shunt→susceptance</td>
<td>$B$</td>
<td>$\frac{1}{Z_{\text{sym}}} \cdot B$</td>
</tr>
</tbody>
</table>

Table 22: Shunt per-unit transformations.
### Table 23: Load per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>demand→active</td>
<td>$P$</td>
<td>$\frac{1}{S_{nom}} \cdot P$</td>
</tr>
<tr>
<td>demand→reactive</td>
<td>$Q$</td>
<td>$\frac{1}{S_{nom}} \cdot Q$</td>
</tr>
</tbody>
</table>

### Table 24: Generator per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>output→active</td>
<td>$P$</td>
<td>$\frac{1}{S_{nom}} \cdot P$</td>
</tr>
<tr>
<td>output→reactive</td>
<td>$Q$</td>
<td>$\frac{1}{S_{nom}} \cdot Q$</td>
</tr>
<tr>
<td>PQ_curve→values[k][0]</td>
<td>$P_k$</td>
<td>$\frac{1}{S_{nom}} \cdot P_k$</td>
</tr>
<tr>
<td>PQ_curve→values[k][1]</td>
<td>$Q_{k_{min}}$</td>
<td>$\frac{1}{S_{nom}} \cdot Q_{k_{min}}$</td>
</tr>
<tr>
<td>PQ_curve→values[k][2]</td>
<td>$Q_{k_{max}}$</td>
<td>$\frac{1}{S_{nom}} \cdot Q_{k_{max}}$</td>
</tr>
</tbody>
</table>

### Table 25: Synchronous condenser per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>output→reactive</td>
<td>$Q$</td>
<td>$\frac{1}{S_{nom}} \cdot Q$</td>
</tr>
</tbody>
</table>

### Table 26: Voltage Level per-unit transformations.

<table>
<thead>
<tr>
<th>GRG name</th>
<th>Parameter</th>
<th>Per-unit transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltage→nominal_value</td>
<td>$V_{nom}$</td>
<td>$\frac{1}{V_{nom}} \cdot V_{nom}$</td>
</tr>
<tr>
<td>voltage→lower_limit</td>
<td>$V_L$</td>
<td>$\frac{1}{V_{nom}} \cdot V_L$</td>
</tr>
<tr>
<td>voltage→upper_limit</td>
<td>$V_U$</td>
<td>$\frac{1}{V_{nom}} \cdot V_U$</td>
</tr>
</tbody>
</table>