How to get started with the "AnnotatedGraph" plugin in ProM?

- The plugin requires an event log array (see the section below how to create such an array) and optionally multiple Disco Process Maps if that is the preferred input model type one is going to use. The suggested steps in starting the plugin:
  - [**Disco Process Map**] Use the EventLogArray object to search for accepting plugins (by clicking "Use resource" after selecting the EventLogArray object) and select the "Compare process models; DiscoProcessMap" plugin. Add the Disco Process Map objects to the input. IMPORTANT: The plugin expects a Disco Process Map for every event log and vice versa and, for every pair of Disco Process Map and event log, the Disco Process Map label has to be equal to the event log label, as they would occur in the ProM workspace when imported as single objects where a prefix "Anonymous log imported from " is also accepted. The latter indicates the absence of a name in the event log file. In such cases the object label in ProM will be similar to "Anonymous log imported from *filename*". Accepted examples of *(disco process map, event log)* label pairs:
    (el1, el1)
    (el1, Anonymous log imported from el1)
  - [**Other**] Use the EventLogArray object to search for accepting plugins (by clicking "Use resource" after selecting the EventLogArray object) and select the "Compare process models; XLog" plugin. A dialog will be shown which requests what type of process model has to be discovered for each of the event logs in the array.
- The plugin will show the *merged annotated graph* where edges are provided with a color that indicates the input model containing the edge. Vertices have their origin in multiple input models and therefore do not show such an indication.
  Selecting an element (vertex or edge) will have as effect that the data contained in the element will be displayed in the table within the *Controls Panel*.

- The *Controls Panel* holds sliders which can be used to reduce the number of elements that are displayed. Filtering is done using a threshold for the frequency of an edge and the minimum frequency (over all input models) for a vertex.
- The *Primitives Panel* provides a list of graphical primitives that can be influenced to change the visualization of the graph. All primitives depend on the same data attribute that is on top of the list. For every element in the graph, the data stored for this selected attribute will be retrieved, a selected function is applied to this data and the result of this operation will determine the visual result: For a vertex, the resulting value will be normalized across all vertices. This means a vertex visualization will be relative to all other vertices in the graph. For an edge, the resulting value will be normalized across only the edges having the same source and target.
- Each setting will be summarized using natural language in righthand corner at the top of the screen. This should help in understanding what is currently visualized.

How to create an event log array in ProM?

There are two options: one can load a predefined array by importing a .ela file or one can first import the separate event logs and combine those to form an array.

A .ela file is a simple text file listing a number of event logs that are located in the same folder as the .ela file. An example .ela file is shown below:

```
eventlog1.xes
eventlog2.xes
eventlog3.xes
```

The imported object is an "EventLogArray".

The other option requires the event logs to be imported manually first. The plugin "Create an event log array" can be used to combine the files into an array. Note that all event logs have to be manually selected after selecting the plugin. The plugin result is again an "EventLogArray".