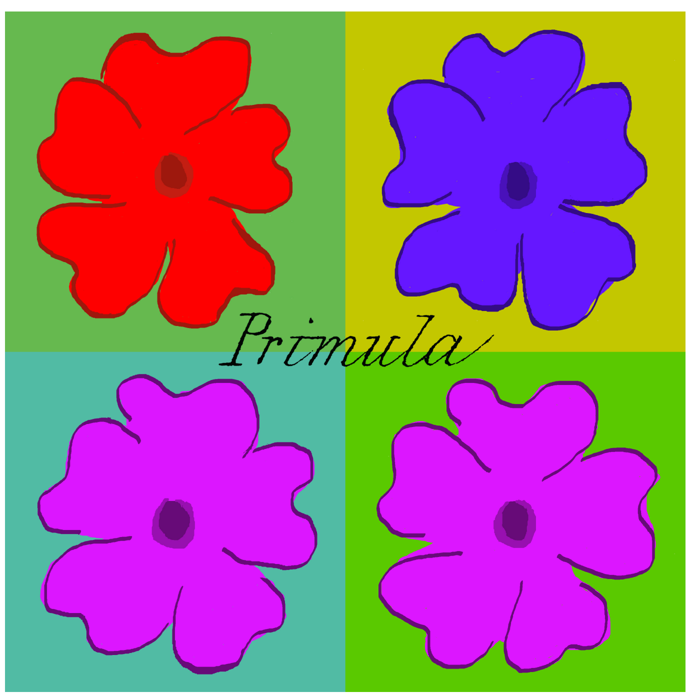
# The *Primula* System: user's guide
# Version 3.0
# Example: `Generalized inference: model-level explainer with MAP inference`


Raffaele Pojer, Manfred Jaeger
Institut for Datalogi, Aalborg Universitet, Selma Lagerløfs Vej 300, 9220 Aalborg Ø
`rafpoj@cs.aau.dk`, `jaeger@cs.aau.dk`


*Primula* homepage: www.cs.aau.dk/∼jaeger/Primula

September 19, 2023



## Introduction

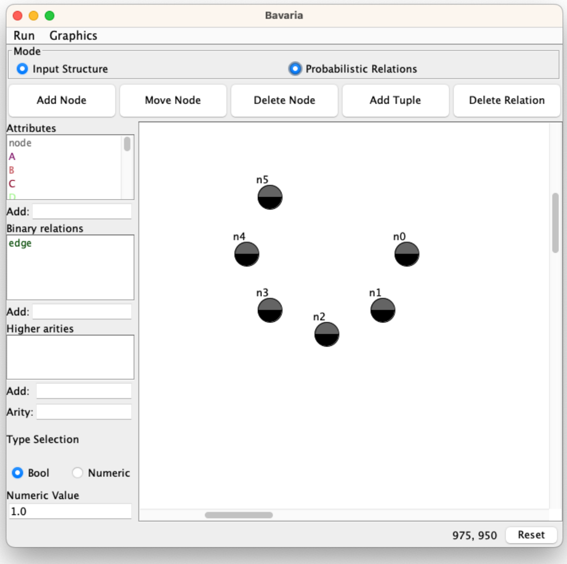
In the field of deep learning, explaining the behavior of a trained model is becoming more and more required. As GNNs are increasingly used in many different fields, explaining techniques have received significant attention. In literature there exists different approaches to explain a GNN model, and most of the work that has been done is focused on instance-level explanation, where the goal is to explain the specific prediction. Not may works have been done on the model-level explanation, where the goal is to explain the model itself. We can define a model-level explainer as a method for construction the *most probable explanation* (MPE) graph for a given class.

In this example we will show how to use *Primula* to compute the MPE graph for a given class, demonstrating how *Primula* can be used as an out-of-the-box GNN explainer. The *.rbn file is a Relational Bayesian Network that is computationally equivalent to a Graph Neural Network, in particular the architecture used is an ACR-GNN. This RBN contains the weights of a trained ACR-GNN model on a synthetic dataset. This dataset is composed by random graphs, where each node can have 7 different types [A, B, C, D, E, F, G]. Positive graphs have inside two specific motifs: one with a node A connected to two node B *(B - A - B)*, and the other is one node C connected to two nodes D *(C - D - C)*. We trained a ACR-GNN to classify the two classes obtaining high accuracy (0.9904), and we export the trained model into a computationally equivalent RBN. Edges in the dataset are not directed, and in the RBN definition direction do not have any importance or meaning for the final results. Using this GNN-RBN combination, we can see with *Primula* which is the most probable *n-node* graph and compute also the probability that this graph has on the model.

### Primula settings

Load the model file `rbn_acr.rbn` and the one of the data file `*.rdef` from the folder (there are 3 files, for the *4/6/8-node* graph).

Select in the *Primula* console Modules:Bavaria to open the graphical data editor. In *Bavaria* press the toggle Probabilistic Relations to view also the attributes of nodes, there will be two color: grey for the "node" attribute, while the black for the constraints. You will see something like this:

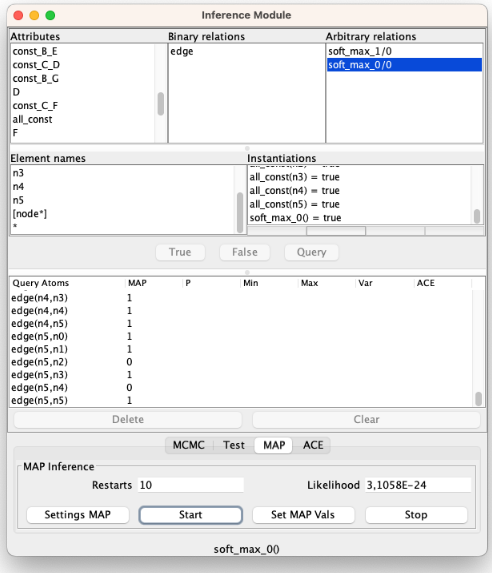



**About the constraints:** With RBNs we can impose some constraint for the graph generation during the MAP inference, those constraints in this case are useful to assign an high probability to having assigned only one attribute of the 7 different types for each node. In this case it will very hard to have graphs with multiple or zero types assigned.

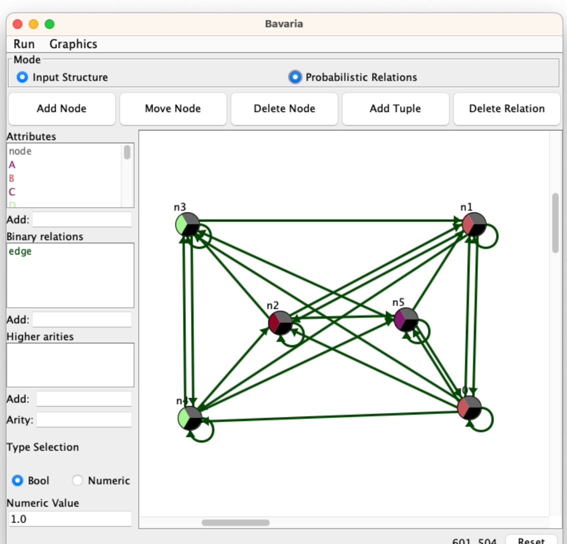
**Setting Queries**
Open the Modules:Inference Module and click the 'Query' button to activate the query mode.

- **Nodes:** Select the blue attribute from the 'Attributes' list and click on the `[node*]` to select all the nodes. In the synthetic example, each node has 7 possible attributes [A, B, C, D, E, F, G].
  In order to do this, click on one attribute (e.g. C) and click on the `[node*]` element in the Element names list. This assigns the selected Attributes to all nodes. Repeat this process for all 7 attributes (Note: the Query button can only be clicked on the first time).
- **Edges:** Assign all possible edge combinations to the nodes by selecting the edge attribute in the Binary relations list, and double-clicking again on the `[node*]` element (this require to be in the "query mode", click on the Query button to enter in this mode).
- **Class:** Set the class for which you want to compute the most probable graph.
  In the compiled RBN, all positive classes are assigned to the Arbitrary relation `soft_max_0/0`, negative to `soft_max_1/0`. Click the True button and then click on the `soft_max_0/0` relation to assign it as the true value.

**Computing MAP**. Start the MAP computation by clicking the MAP button and then clicking the Settings MAP button. A new window will open, allowing you to set the number of restarts you want (-1 to stop manually or another value, e.g., 10). Click the Start button in the Inference Module and wait for the restarts to complete or manually stop the computation. When the computation finishes, press the 'Set MAP Vals' button to apply the computed values to the graph.




**Visualizing the results**. View the computed values from Bavaria. Save these results in an .rdef file by clicking on the *Primula* main interface's Save Data option under the Run menu. In Bavaria is it possible to see the color of the attributes to distinguish the different types of nodes. Otherwise is it possible to see the computed values in the Inference Module. In the image below the two motifs for the positive class are being perfectly generated.



## Extra: additional analysis

It is possible to compute the probability that this generated graph for the model directly on the *Primula* interface. The .rbn file represent the model and we can use the MCMC inference module to estimate the probability that the generated graph has on the GNN model with few decimals of error.

Open the saved .rdef file computed with the MAP inference with a text editor. This file is an XML-like file with all the attributes specified for the graph. In the `<ProbabilisticRelsCase>` there are written all the assignments for the elements in the graph. Look for for the relation `<d rel="soft_max_0" args="()" val="true"/>` and delete that from the .rdef file. In this way we are not assign the class to the graph and it will be possible to compute the true probability (otherwise it will be as 1.0 for the assignment).
Load the modified .rdef file computed with the MAP inference into *Primula* with the same .rbn of the computation of before. Open the Inference Module and click on the MCMC button. Click on the Query button to query which class we want to select and click to the textttsoft_max_0/0 class in the Arbitrary relations (it is possible also to click on both the classes to have also the probability of the other class). Click on the Start button and wait some seconds, the Sample Size will increase and in the Query Atoms it will be possible to see the computed probability under the P column.