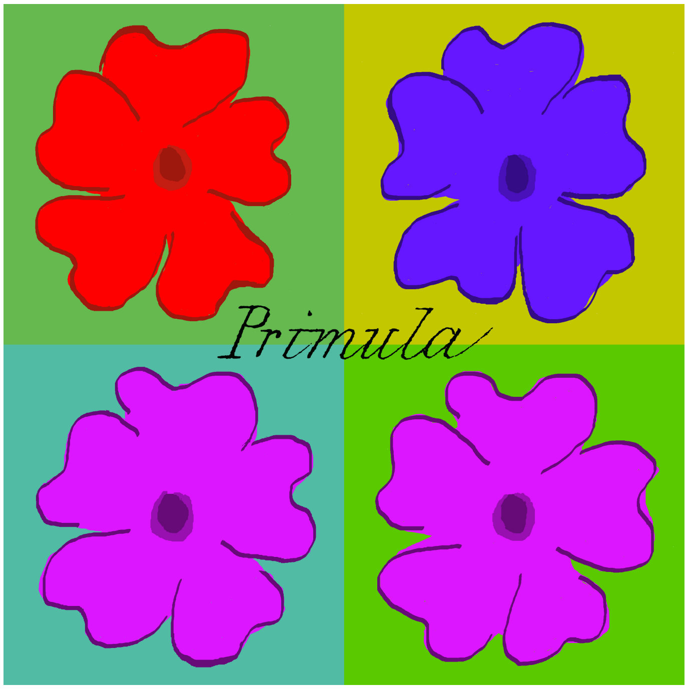
The *Primula* System: user's guide Version 3.0 Example: `Generalized inference: compute the graph structure`

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

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

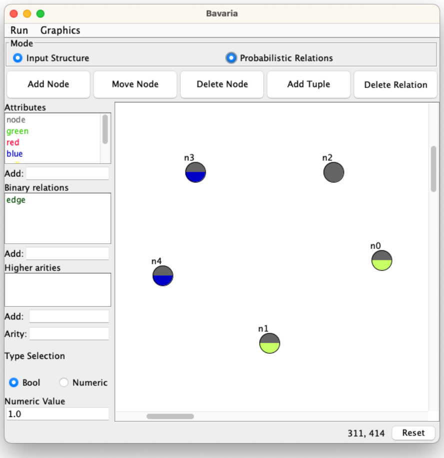
September 8, 2023



Generalized inference: compute the graph structure

Load the model file `rbn_acr.rbn` and the data file `alpha1-edge.rdef` from the folder.

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, blue color is for the blue attribute, lime color is for the α_1 attribute, while no colors or grey means nodes without any assignments. You will see something like this:



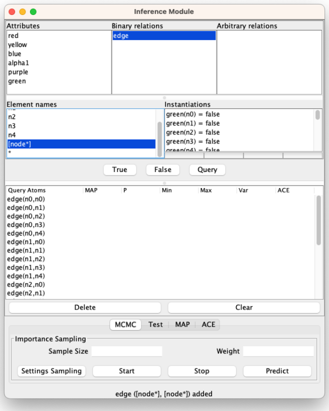
The *Bavaria* window displays the relational structure contained in `alpha-blue.rdef`.

NB: With RBNs and in Primula all the graphs are directed, as you can see also in this graphs edges have direction, but the RBN we have just imported do not take in consideration the direction of the edges.

The model `rbn_acr.rbn` is a RBN which represents an ACR-GNN model. The model is a single-layer ACR-GNN, and it was trained on a synthetic dataset by taking inspiration from the paper of The Logical Expressiveness of Graph Neural Networks of Barceló et al (2019). For more details about the model and the dataset, see the paper *Generalized Reasoning with Graph Neural Networks by Relational Bayesian Network Encodings* from Pojer et al (2023). For a more easy reading, we report here the formula of the α_1 logical classifier we have adopted in this example:

$$\alpha(x) := \exists^{[2,3]} y(\mathrm{Blue}(y) \wedge \neg \mathrm{edge}(x, y)). \tag{1}$$

Generate the graph structure using MAP inference. Open the Modules:Inference Module and press the 'Query' button to activate the query mode. Now select the edge attribute from the 'Binary Relations' list and double-click on the `[node*]` to query the edge attribute for all the nodes. You will see something like this:



Inference Module

Attributes

red
yellow
blue
alpha1
purple
green

Binary relations

edge

Arbitrary relations

Element names

n2
n3
n4
[node*]
*

Instantiations

green(n0) = false
green(n1) = false
green(n2) = false
green(n3) = false
green(n4) = false

Query Atoms	MAP	P	Min	Max	Var	ACE
edge(n0,n0)						
edge(n0,n1)						
edge(n0,n2)						
edge(n0,n3)						
edge(n0,n4)						
edge(n1,n0)						
edge(n1,n1)						
edge(n1,n2)						
edge(n1,n3)						
edge(n1,n4)						
edge(n2,n0)						
edge(n2,n1)						

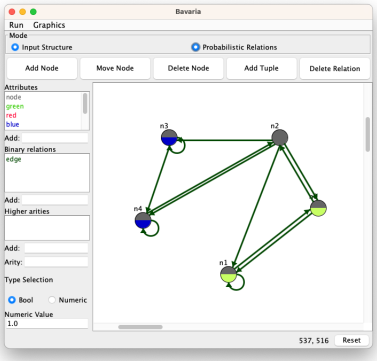
Importance Sampling

Sample Size

Weight

edge ([node*], [node*]) added

Configure the MAP inference tool to compute the graph structure. Select the 'MAP' at the bottom of the window and press the 'Setting MAP' button. A new window will appear. Enter the number of restarts you want to perform in the first text box (e.g. -1 for infinite restarts, or 10 as in the paper examples). After that, click on the 'Start' button to start the MAP inference. After a while, the MAP inference will stop to the restarts number you have set, otherwise press the 'Stop' button to stop the MAP inference. Assign the results of the MAP inference to the graph using the 'Set MAP Vals' button. In *Bavaria* you will see the results with the new graph structure.



Bavaria

Mode

● Input Structure ● Probabilistic Relations

Attributes

node
green
red
blue

Add:

Binary relations

edge

Add:

Higher arities

Add:

Arity:

Type Selection

● Bool ○ Numeric

Numeric Value

1.0

537, 516