# **M2 - Big Data Management and Analytics**

### **Decision Modelling – Practical 1 - Preferences as Binary Relations**

#### Rishika Gupta

This document adds some comments to the code of the practical works.

* Requirements.txt file contains all the libraries that need to be imported for the code.
* Helping functions are defined for converting between csv, matrix and graph.
  + The input to the code is of the format .xlsx. It is then converted to .csv for further processing.
  + Matrix is of the format numpy.matrix.
  + The graph is of the format networkx.graph (a Directed Graph in this case).
* Within the code, the binary relations are converted to matrices and graphs whenever required.

1. **Python function converting and excel file (.xls or .xlsx) to a .csv file**

Input Parameters: An excel file

Output: No output, the csv file is saved to the current working directory directly.

1. **Python function for a graphical representation of the matrix using networkx and matplotlib**

Input Parameters: A numpy matrix

Output: A networkx graph

In this function, checks are made to see if the binary relation is of the form m x m and also the value of m to lie in the range [3,20]. If both hold true, matrix is converted to graph and visualised.

1. **Python function CompleteCheck testing if a binary relation is complete**

Input Parameters: A numpy matrix

Output: Boolean, indicating if it is complete or not.

Example:

[[1, 1, 1],

[0, 1, 1],

[0, 0, 1]]

1. **Python function ReflexiveCheck testing if a binary relation is reflexive**

Input Parameters: A numpy matrix

Output: Boolean, indicating if it is reflexive or not.

Example:

[[1, 0, 0],

[0, 1, 1],

[0, 0, 1]]

1. **Python function AsymmetricCheck testing if a binary relation is asymmetric**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is asymmetric or not.

Example:

[[1, 1, 0],

[0, 0, 1],

[0, 0, 1]]

1. **Python function SymmetricCheck testing if a binary relation is symmetric**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is symmetric or not.

Example:

[[1, 0, 0],

[0, 1, 1],

[0, 1, 1]]

1. **Python function AntisymmetricCheck testing if a binary relation is antisymmetric**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is antisymmetric or not.

Example:

[[1, 1, 0],

[0, 1, 0],

[0, 0, 1]]

1. **Python function TransitiveCheck testing if a binary relation is transitive**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is transitive or not.

Example:

[[1, 1, 1],

[0, 0, 1],

[0, 0, 0]]

1. **Python function NegativeTransitiveCheck testing if a binary relation is negative transitive**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is negative transitive or not.

Example:

[[0, 1, 1],

[0, 0, 0],

[0, 0, 0]]

1. **Python function CompleteOrderCheck testing if a binary relation is a total order**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is a complete order or not.

Example:

[[1, 1, 1],

[0, 1, 1],

[0, 0, 1]]

1. **Python function CompletePreOrderCheck testing if a binary relation is a preorder**

Input parameters: A numpy matrix

Output: Boolean, indicating if it is a preorder or not.

Example:

[[1, 1, 1],

[0, 1, 1],

[0, 0, 1]]

1. **Python function StrictRelation returning the asymmetric part of a binary relation**

Input parameters: A numpy matrix

Output: A numpy matrix

Example:

Input:

[[1, 0, 0],

[1, 1, 1],

[0, 0, 1]]

Output:

[[0, 0, 0],

[1, 0, 1],

[0, 0, 1]]

1. **Python function IndifferenceRelation returning the symmetric part of a binary relation**

Input parameters: A numpy matrix

Output: A numpy matrix

Example:

Input:

[[1, 1, 0],

[0, 1, 1],

[0, 1, 1]]

Output:

[[1, 0, 0],

[0, 1, 1],

[0, 1, 1]]

1. **Python function Toplogicalsorting returning a topological sorting of a given binary relation without cycles**

In the code for this function, it is assumed that there are no ties in ranking.

Input parameters: A numpy matrix

Output: A numpy matrix

Example:

Input:

[[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0],

[0, 0, 0, 1, 0, 0],

[0, 1, 0, 0, 0, 0],

[1, 1, 0, 0, 0, 0],

[1, 0, 1, 0, 0, 0]]

Output: 4 -> 5 -> 2 -> 3 -> 0 -> 1

1. **Creating test functions for all the above-mentioned functions are run for the new input numpy matrix B**

Input:

B = [[1, 1, 1, 1, 1, 1],

[0, 1, 1, 1, 1, 1],

[0, 0, 1, 1, 1, 1],

[0, 1, 1, 1, 1, 0],

[0, 0, 0, 1, 1, 1],

[0, 0, 0, 0, 1, 1]]

Output:

* B is not complete.
* B is reflexive.
* B is not asymmetric.
* B is not symmetric.
* B is not antisymmetric.
* B is not transitive.
* B is not negative transitive.
* B is not a complete order.
* B is not a complete preorder.
* Strict relation of B is:

[[0 1 1 1 1 1]

[0 0 1 0 1 1]

[0 0 0 0 1 1]

[0 0 0 0 0 0]

[0 0 0 0 0 0]

[0 0 0 0 0 0]]

* Indifference relation of B is:

[[1 0 0 0 0 0]

[0 1 0 1 0 0]

[0 0 1 1 0 0]

[0 1 1 1 1 0]

[0 0 0 1 1 1]

[0 0 0 0 1 1]]

* Since B has cycles, it cannot have a topological sorting for this relation.