

# Python coding with graphs

Prof. Ivano Notarnicola

Department of Electrical, Electronic, and Information Engineering Alma Mater Studiorum Università di Bologna

ivano.notarnicola@unibo.it

Distributed Autonomous Systems M A.A. 2024-2025

### Graph theory recap



A **digraph** (directed graph) is a pair G = (I, E) where

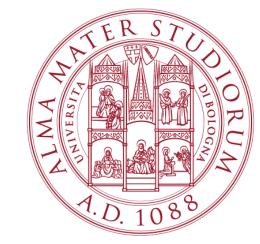
- $I = \{1,...,N\}$  is a collection of elements, called **nodes**
- $E \subset I \times I$  is a set of ordered node pairs, called *edges*

An edge from node i to j is a pair denoted as (i, j)

In Python we can use the package *networkx* <a href="https://networkx.org/">https://networkx.org/</a>
import networkx as nx



#### Networkx: graph creation



An undirected graph in Python can be instantiated via

$$G = nx.Graph()$$

A node, e.g., with index 0, can be added to the graph G using

G.add\_node(0)

G.add\_nodes\_from([1, 2, 3])

An edge, e.g., between node 1 and 2, can be added to the graph via

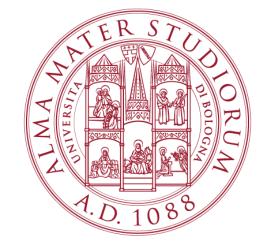
 $G.add_edge(1, 2)$ 

**G.add\_edges\_from([(1, 2), (1, 3)])** 

We can draw a graph with

nx.draw(G) or nx.draw\_circular(G, with\_labels=True)

#### Networkx: create a directed graph



A directed graph can be created via

directedG = nx.DiGraph()

We can obtain the list of neighbors of a node, e.g., of node 0, using

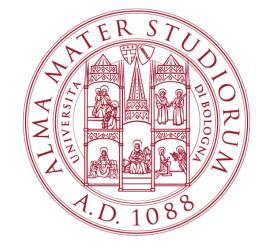
N\_i = nx.neighbors(directedG, 0)

(Its explicit casting as a **list** is useful to use it)

Extract the adjacency matrix as a numpy array using

Adj = nx.adjacency\_matrix(directedG).toarray()

## Networkx: compute the Laplacian of a graph



We can compute the Laplacian matrix via

Alternatively, we can resort to its definition and manually compute it

First, let I\_N be the identity matrix of order N, then compute the degree matrix via

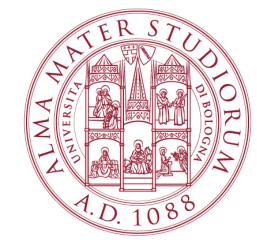
$$degree = np.sum(Adj + I_N, axis = 0)$$

$$D_IN = np.diag(degree)$$

The in-Laplancian matrix is finally given by

$$L_IN = D_IN - Adj.T$$

#### Networkx: predefined graphs



We can create a **path** graph via

We can create a *cycle graph* via

We can create a **star graph** via

We can also create more general random graph as, e.g., an Erdős-Rényi graph as

#### Stochastic matrices



#### A non-negative square matrix A is

- row stochastic if its rows sum up to 1
- column stochastic if its columns sum up to 1
- doubly stochastic if it is both row and column stochastic

What happens to the linear averaging algorithm when

- the matrix is column stochastic but not row stochastic?
- the matrix is row stochastic but not column stochastic?