Random Graphs

Abdoulaye GASSAMA
Sen ZHOU

February 24, 2026

Contents

Abstract

Definition 1. A graph $G = (V, E)$ is a collection of elements called vertices connected by edges. The set of vertices is denoted by V (from "vertices"), and the set of edges is denoted by E (from "edges"), where $E \subset V \times V$. We denoted the set of graphs $\mathcal{G}$.

Definition 2. Let $n \in \mathbb{N}$ and $p \in [0,1]$. A random graph $G \in \mathcal{G}(n,p)$ is defined as $G = (V, E)$ where: $V = \{1, ..., n\}.\{i,j\} \in \binom{V}{2}$, $\mathbb{P}(\{i,j\} \in E) = p$.

1 Simulation of Graph $\mathcal{G}(n,p)$

In this section the objective is to simulate a random graph carried out by Python.For this we will need of the library networkx which give access to some useful functions.

1. Initialisation of parameters

```
n = 10 # number of vertices
p = 0.5
nb_graph = 10**4
graph_list = []
```

2. Function which generates an oriented random graph (i < j)

```
def generate_oriented_graph(n, p):
    G = nx.DiGraph()
    G.add_nodes_from(range(1, n+1))

    for i in range(1, n+1):
        for j in range(i+1, n+1):  # i < j
            if np.random.rand() < p:
                G.add_edge(i, j)

    return G
```

3. Loop which generates a list of graph

```
for i in range(nb_graph):
    G = generate_oriented_graph(n, p)
    graph_list.append(G)
```

4. Computation of $\frac{\#\{\text{graphs where } 1 \to n\}}{\#\{\text{simulations}\}}$

```
graph_1_n = 0
for i in range(nb_graph):
    if (nx.has_path(graph_list[i], 1, n)):
        graph_1_n += 1
```