SNOM HW1 - Theoretical Question

Michele Luca Puzzo 1783133

April 2022

1 Problem 1

Consider a graph created according to the Erdős-Rényi $G_{n,p}$ random-graph model.

1.3

I have n nodes. The total number of possible edges that can be formed is equal to the simple combinations of n elements taken two by two: $\binom{n}{2}$. The other parameter of the model is $p \in [0,1]$ which is the probability that an edge exists, so it is interpreted as the probability of "success" that an edge is formed.

In conclusion the number of edges m is a random variable distributed as:
$m \sim Binom\left(\binom{n}{2}, p\right)$ and so the expected number of edges is: $\mathbb{E}[\mathbf{m}] = \binom{\mathbf{n}}{\mathbf{2}} \cdot \mathbf{p}$

For a large number of nodes the expected number of edges will be very close to $n \cdot p$.

1.1

From the previous point I know that $m \sim Binom\left(\binom{n}{2}, p\right)$ and so:
$\mathbb{P}(\text{"a particular graph with } m \text{ edges"}) = p^m(1-p)^{\binom{n}{2}-m}$

Taking three edges and three nodes I obtain a triangle so firstly I am looking for a graph with exactly three edges. Moreover if I have n nodes the way to pick three of them is equal to the simple combinations of n elements taken three by three: $\binom{n}{3}$. Combining these two considerations I have:
$\mathbb{P}(\textbf{"graph contains exactly only one triangle"}) = \binom{\mathbf{n}}{\mathbf{3}} \cdot \mathbf{p^3(1-p)^{\binom{n}{2}-3}}$ with $n \geq 3$

1.2

Firstly to build a line with n nodes I need exactly n-1 edges, so similarly as before I have $\mathbb{P}(\text{"a particular graph with } \textit{n-1} \text{ edges"}) = p^{n-1}(1-p)^{\binom{n}{2}-(n-1)}$.
Then I have found that the total number of possible graph made by n nodes connected in a line is $\frac{n!}{2}$: I have considered the number of way to arrange n nodes in a line is equal to simple provisions with n elements: n! (in this case I have to use provisions and not combinations because the order of nodes matters). It is important to notice that a graph has two provisions corresponding to itself indeed each line can be written from right to left and the other way around but it is the same graph. For example 1-2-3 is the same graph of 3-2-1, 2-1-3 is the same graph of 3-1-2 and so on. In conclusion, combining these two results I have:
$\mathbb{P}(\textbf{"all the n nodes are connected in a line"}) = \frac{\mathbf{n!}}{\mathbf{2}} \cdot \mathbf{p^{n-1}(1-p)^{\binom{n}{2}-(n-1)}}$ with $n \geq 2$

2 Problem 3

As notation I have used:

- V: set of nodes of the network.
- S: set of nodes that contains people that I have decided to follow.
- k: budget or the maximum number of people that I can follow.
- N_S: set of nodes that are neighbours of at least one node contained in S. N_v: set of nodes that are neighbours of v.
- $f(S) = |S \cup N_S|$ It is the number of nodes in S and nodes reachable from S in one hop.

The aim of my algorithm is:

$$\begin{aligned} &\text{maximize} && f(S) \\ &\text{subject to} && S \subseteq V,\ |S| \leq k. \end{aligned}$$

Initially to solve this problem I have thought to follow the k people who have the k highest degree in the network. Then I have reckoned this heuristic not adequate because in the case in which people with the highest degree follow each other I would obtain a bad solution. In the counterexample that I have created in the figure if k=2 I should follow for example the people 1 and 2 because they have both degree = 3, but in this way I will know just four users. On the other if I followed 1 and 5, that has degree = 2, I would know seven users!

Figure 1: Counterexample

For this reason I have slightly modified my initial idea. As first node I take the node with the highest degree as before. But from the second node on I will take the node that has as maximum the difference between its degree and the number of its neighbours that are already "known" from previous selected nodes. In this way I will prevent me to take a node that have a lot of friends that are friends of people that I have already followed, but I will take or "follow" nodes that have a lot of friends still unknown to me. So at each step maximizing my function $|S \cup N_S|$ adding a node v means to find a node v that maximizes $degree(v) - (|N_S \cap N_v|)$. To do this whenever I have to choose a node I have to scan all the nodes not previously taken.

Algorithm 1 Monitoring

```
function MONITOR(Network, k)
    S ← ∅
    N_S ← ∅
    while |S| ≤ k do
        find v ∈ V\S : max(f(S ∪ {v})) = max(degree(v) − (|N_S ∩ N_v|))
        S ← S ∪ {v}
        N_S ← N_S ∪ N_v
    end while
    return S
end function
```

The number of steps of my algorithm is $\mathcal{O}(|V| \cdot k)$ because I have a while cycle made by k iterations and in each iteration I scan each node of the graph.

A function $f : 2^V \to \mathbb{R}$ is submodular if $\forall\ S, T \subseteq V$ with $S \subset T$ and $v \in V \backslash T$ we have: $f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$.
My $f(S)$ is a submodular function because:

$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$
$\Rightarrow |S \cup N_S| + 1 + degree(v) - |N_S \cap N_v| - |S \cup N_S| \geq |T \cup N_T| + 1 + degree(v) - |N_T \cap N_v| - |T \cup N_T|$
$\Rightarrow |N_S \cap N_v| \leq |N_T \cap N_v|$

This is true because $S \subset T$ so $N_S \subseteq N_T$. Moreover in class I have seen that reachability is a submodular function.

There is a theorem that claims that this greedy algorithm gives a $\left(1 - \frac{1}{e}\right) \approx 0.63$ approximation.

3 Problem 4

First Example

In the following figure I provide an example of a graph in which the densest subgraph is $D = \{1, 2, 3, 4\}$ which coincides with the sparsest cut S. The partition of the graph is marked in the figure from the color of nodes. Indeed D is a clique of size four and it maximizes $\frac{|E \cup (D \times D)|}{|D|} = \frac{E(D)}{|D|} = \frac{6}{4} = 1.5$.

At the same time D is also the sparsest cut indeed it minimizes $\frac{|E \cup (S \times (V \backslash S))|}{min\{|S|, |V \backslash S|\}} = \frac{1}{min\{4,4\}} = 0.25$. Choosing D the size of the cut is one, just the bold edge in the figure (1,5) that joins the two subgraphs, S and $V \backslash S$.

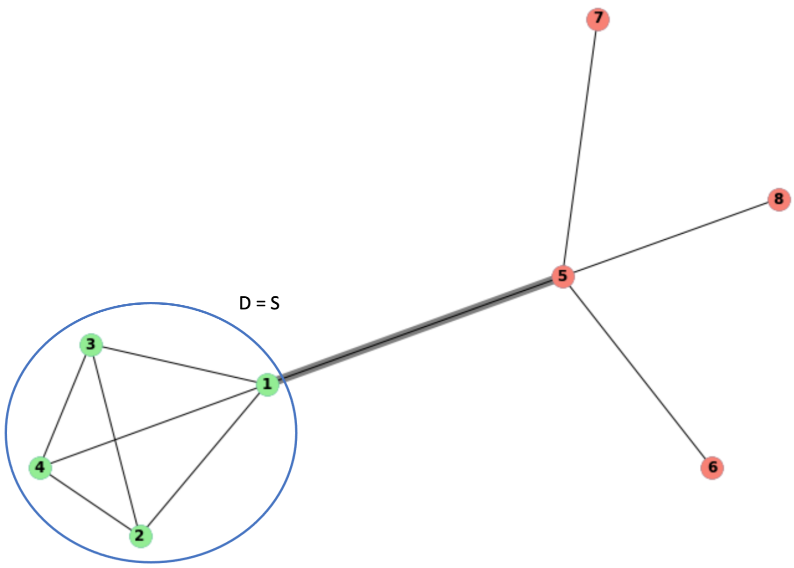

Figure 2: Example in which D = S

Second Example

In this second example the densest subgraph is $D = \{1, 2, 3, 4, 5\}$ because it makes $\frac{|E \cup (D \times D)|}{|D|} = \frac{E(D)}{|D|} = \frac{8}{5} = 1.6$ that is the maximum value that is reachable (for example clique $\{1, 2, 3, 4\}$ reaches just $\frac{6}{4} = 1.5$). On the other hand the sparsest cut is $S = \{1, 9, 10, 11, 12\}$ because it minimizes $\frac{|E \cup (S \times (V \backslash S))|}{min\{|S|, |V \backslash S|\}} = \frac{2}{min\{5,12\}} = 0.4$ The partition of the graph is marked in the figure from the color of nodes while the bold edges (1,2), (2,3) represent the cut. D is not contained neither in S nor in $V \backslash S$ indeed the node $1 \in S$ while $2, 3, 4, 5 \in V \backslash S$.

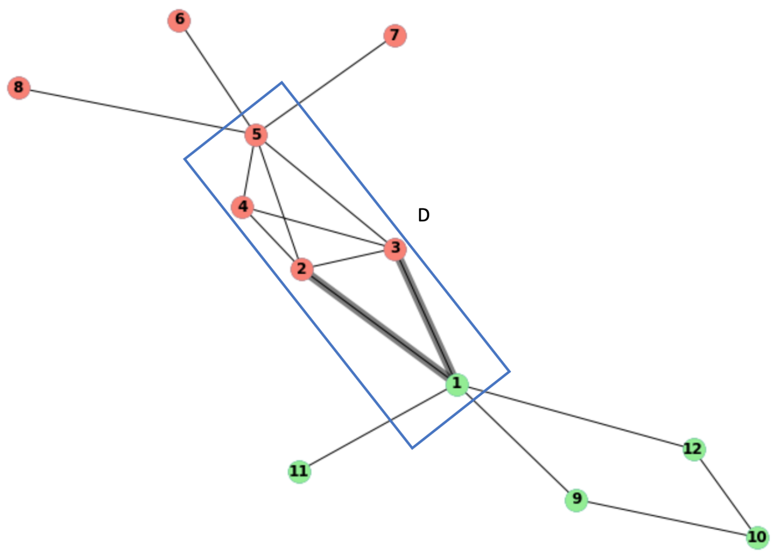

Figure 3: Example in which $D \notin \{S, V\backslash S\}$