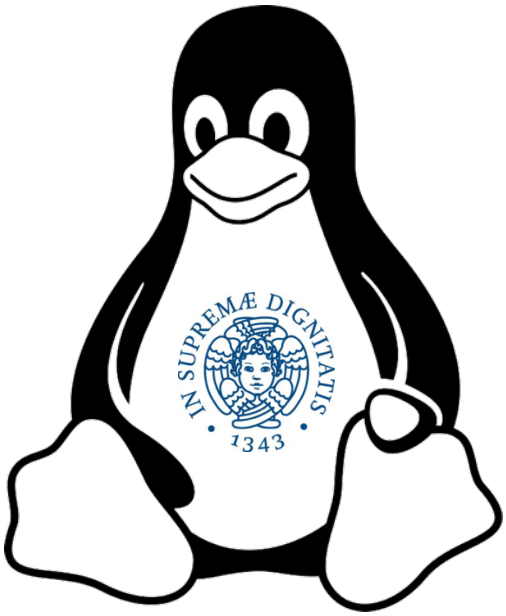



Analisi delle Reti Sociali

# Creazione di una rete basata sulla diffusione dei sistemi GNU/Linux

A.A. 2015-2016

**Alessandro Romano** (537128)

**Tommaso Furlan** (538049)

# Introduzione

L'obiettivo principale del nostro lavoro è la creazione di una rete sociale basata sulla diffusione e l'utilizzo dei sistemi operativi GNU/LINUX nel mondo. I dati sono stati raccolti dal portale Linux Counter[1], il quale sotto forma di censimento colleziona le informazioni relative alle macchine grazie alla collaborazione diretta degli utenti.
Gli amministratori del sito ci hanno fornito un drop del database contenente tutti i dati raccolti fino al 10/03/2016, dove ogni record fa riferimento ad un'unica macchina. Abbiamo selezionato i seguenti attributi, ritenuti utili per l'analisi:

- *numCores* – numero di core del processore
- *kernel* – versione del kernel
- *class* – classe di appartenenza
- *cpu* – modello del processore
- *country* – nazionalità
- *architecture* – architettura del processore
- *distribution* – distribuzione

Segue una fase di preparazione del dataset in cui si procede alla correzione di errori ed alla rimozione di record con attributi nulli, quindi ad una normalizzazione dei valori numerici nel range [0,1]. Con l'obbiettivo di evitare casi di inconsistenza, si effettuano altrettante operazioni di preparazione che non sono direttamente collegate allo scopo di questo documento, pertanto ne viene omessa la descrizione.

Gran parte del lavoro si è focalizzato sulla costruzione della rete, in modo da ottenere un grafo che la rappresentasse il più possibile.

In fine si presentano le statistiche di analisi della rete in questione e le tecnologie utilizzate in ogni fase del progetto con alcuni dettagli implementativi.

# Costruzione della rete

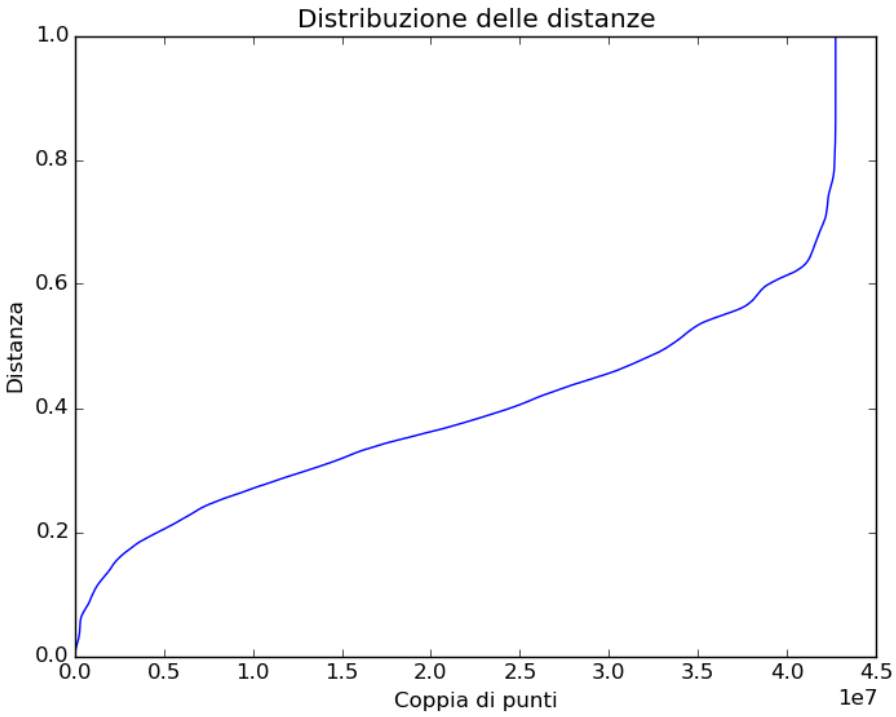
Partiamo da un dataset contenente circa 9000 nodi, pertanto procediamo alla costruzione della rete calcolando, come prima cosa, la distanza tra ogni nodo. Ogni macchina è descritta dagli attributi elencati precedentemente, quindi definiamo una formula adeguata per questa operazione, valorizzando il più possibile ogni attributo e limitando la perdita di informazione.
Siano M1 ed M2 due macchine, definiamo la distanza tra queste come:

$$\mathbf{dist(M1,M2)} = numCores\_dist() + kernel\_dist() + class\_dist() + country\_dist() + architecture\_dist() + distro\_dist()$$

[1] www.linuxcounter.net/

Dove:

- numCores_dist() calcola la differenza in valore assoluto tra i valori che quantificano il numero di core di un processore
- kernel_dist() calcola la differenza in valore assoluto tra due versioni del kernel linux. Ad esempio, “3.2.1” e “4.1.1” diventano |321-411|=90
- class_dist() calcola la distanza tra due classi di macchine, verificando se si trovano in una categoria simile. Ad esempio, la classe “smartphone” appartiene alla stessa categoria della classe “single-board PC”.
- country_dist() calcola la distanza in km tra due nazioni
- architecture_dist() calcola la distanza tra la stringhe che descrivono l'architettura di due macchine. Ad esempio, alle architetture “i386” e “i486” è assegnato uno score di distanza dello 0.1
- distro_dist() calcola la distanza tra due distribuzioni utilizzando un albero che ne descrive le derivazioni. Ad esempio, “Ubuntu” e “Linux Mint” sono entrambe derivate della distro “Debian”




Per ogni coppia di macchine si applica la funzione di distanza appena descritta ottenendo un grafo completo, pesato e non orientato. La rete definitiva, oggetto di analisi, viene costruita eliminando gli archi per cui la distanza tra i nodi supera un valore soglia fissato. Questa soglia è definita empiricamente, ispezionando la curva che descrive le distanze tra ogni coppia, ordinate partendo dalla più bassa. Sull'asse delle ascisse vengono riportati gli archi e sulle ordinate la loro lunghezza. Quindi si è tagliato tutti quelli per cui la distanza tra i nodi è eccessivamente alta, paragonata a quella calcolata tra tutti gli altri nodi del grafo. Questo fenomeno si manifesta esattamente nell'ultima parte della curva, dove le distanze assumono valori altissimi molto rapidamente. Il valore delle ordinate corrispondente a quel punto è esattamente la soglia che utilizziamo per il taglio.

Per questioni tecniche, il grafo ottenuto è troppo grande per essere analizzato, quindi si taglia ulteriormente utilizzando una soglia più bassa pari allo 0.15. Sempre grazie ad un'analisi della *Degree Distribution* dell’intera rete, ipotizziamo che tutti gli archi per cui lunghezza cresce in modo costante non forniscono informazioni molto discriminanti.
Si ottiene un grafo di **9184 nodi** e ***2213198*** **archi.**

# Network Analysis

## Basic Natural Network Measures (Undirected Graph)

In questa sezione vengono spiegate le principali misure delle reti, utili ad approfondire le analisi effettuate a livello pratico.
Riportiamo di seguito le principali metriche:

- **Numero nodi** (N): numero totale dei nodi
- **Numero archi** (L): numero totale degli archi
- **Shortest path**: cammino minimo tra due nodi della rete, ovvero quel percorso che collega due macchine linux minimizzando il numero di connessioni (edge) tra essi
- **Diameter** ($d_{MAX}$): cammino più lungo tra due nodi della rete, cioè la distanza massima tra ogni coppia di nodi
- **Average Path Lenght** (<d>): misura che esprime la lunghezza media relativa agli shortest path
- **Degree**: dato un nodo i, determina il numero di nodi collegati direttamente a tale nodo
- **Average Degree** (<k>): degree medio dei nodi della rete

$$< k > = \frac{2L}{N}$$

- **Degree Distribution**: rappresenta la distribuzione di probabilità dei gradi della rete.

$$P(k) = \frac{N_k}{N}$$

  Per ogni possibile grado k, viene indicata con $N_k$ la frazione dei nodi con tale grado e con P(k) la possibilità che un nodo scelto in modo causale abbia grado k

- **Density**: fff è definita come il rapporto tra il numero di archi presenti e il numero di possibili archi che ci potrebbero essere.

$$D = \frac{2L}{N(N-1)}$$

- **Clustering Coefficient**: vuole quantficare l'esistenza di una comunità all'interno della rete. In pratica questa misura permette di rilevare l'importanza di un nodo.

$$C_i = \frac{2e_i}{k_i\,(k_i - 1)}$$

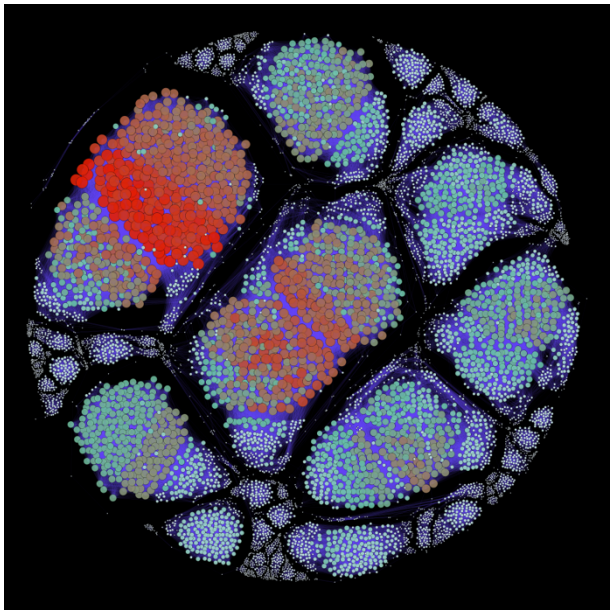
dove $e_i$ indica il numero di link che connettono direttamente i nodi vicini di i, mentre $k_i$ sarà il numero di vicini di i.

$$C = \frac{\sum_1^n C_i * w}{\sum_1^n w_i}$$

dove $C_i$ è il coefficiente locale, mentre $w_i$ è il peso del nodi.

## Linux Network Analysis

| ***Linux Network*** | |
|---|---|
| *N* | *9184* |
| *L* | *2213198* |
| *Connected Components* | *46* |
| *AVG Degree* | *482* |
| *Density* | *0.05245* |
| *Clustering Coefficient* | *0.694312* |
| *Diameter* | $+\infty$ |
| *Average Path Lenght* | $+\infty$ |



Attraverso la libreria di NetworkX di python e alle funzioni disponibili, sono state calcolate le metriche riportate precedentemente, applicandole alla nostra rete.
Le prime due misure saranno quelle principali relative al numero di nodi e archi già citate precedentemente e che ci danno l'idea della dimensione della rete e del grado di collegamento dei nodi, quest'ultimo anche grazie *all'Average Degree*.
Il *Clustering Coefficient* come già citato precedentemente è un indice che misura la densità locale della rete, ovverò la probabilità che due nodi adiacenti siano connessi fra loro, nel nostro caso avremo una probabilità del 69% che come si può immaginare è superiore ai valori riscontrabili di solito in una rete reale.
Avremo infine il *Diameter* e *l'Average Path Lenght*, rispettivamente la distanza massima tra due nodi e quella media, il cui valore però sarà pari a $+\infty$ a causa del fatto che la rete è formata da 46 componenti connesse.

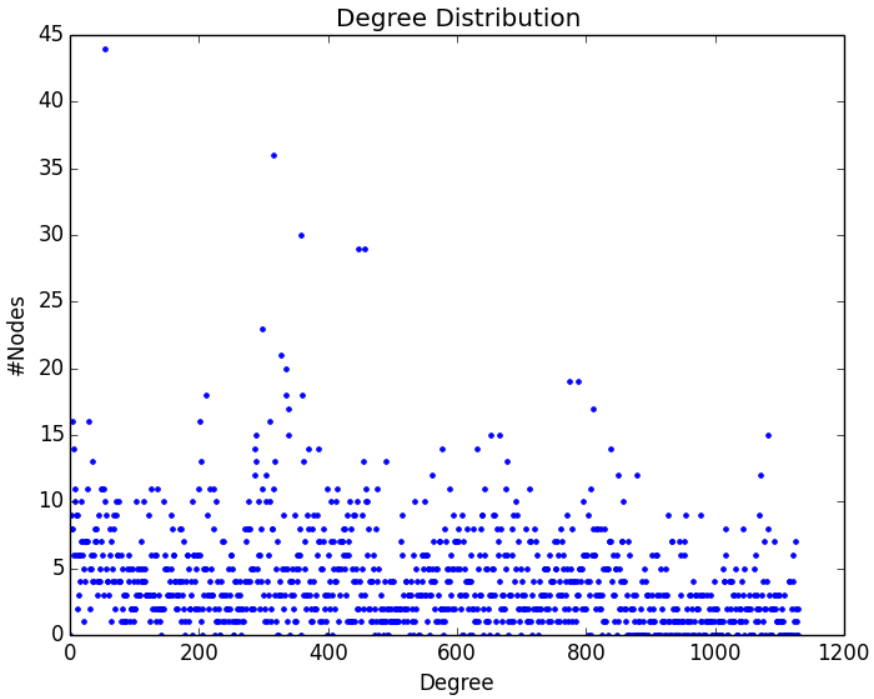
Procederemo quindi ad isolare la **Giant Component principale**, eseguendo su di essa il resto dell'analisi.

| *Giant Component Linux Network* | |
|---|---|
| *N* | *4786*[2] |
| *L* | *1829566* |
| *Connected Components* | *1* |
| *AVG Degree* | *765* |
| *Density* | *0.159780* |
| *Clustering Coefficient* | *0.647806* |
| *Diameter* | *10* |
| *Average Path Lenght* | *2.330747* |

Isolata la componente Giant, potremo notare diverse somiglianze con le metriche della rete Linux completa. Il numero dei nodi che la compongono è il 52% dell'intera rete, con un *Average Degree* che cresce fino a circa 765, infatti come si può vedere la *Density* è molto alta.
Ci soffermiamo su una metrica come *Average Path Lenght* che si collega ad una proprietà di grande importanza nelle reti sociali come il grado di separazione che hanno tra loro i nodi della rete. Come si può vedere nella tabella soprastante, il valore è pari a 2.330747, ciò significa che mediamente un nodo può raggiungere un qualsiasi altro nodo in 3 "passi". Questo fenomeno è detto **Small World**.

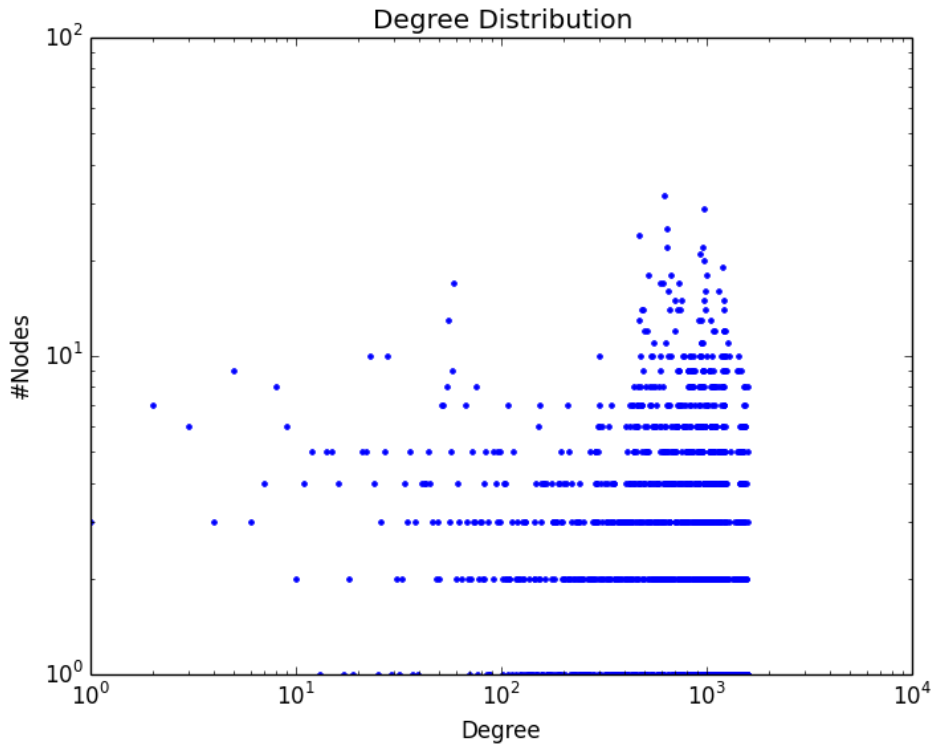
## Degree Distribution

Analizziamo la Degree Distribution, che ci permetterà di comprendere meglio la forma della rete. L'istogramma seguente, permette di capire come il grado (= il numero di neighbors) sia distribuito sui nodi della rete.




[2] Seppur la consegna del progetto richiedesse almeno un Giant di 5000, per limiti computazionali il risultato attuale è stato il massimo ottenibile.

La distribuzione non presenta un particolare addensamento su un grado in particolare, ed è evidente come risulti difficile applicare il concetto di **hub**, ma soprattutto attraverso questo andamento abbastanza difficile da identificare, siamo alquanto lontani dal principio della **Power Law**.
Con lo scopo di rafforzare questa analisi, è stato sviluppato lo stesso diagramma ma in scala logaritmica.




## Centrality Measures

Le misure di Centrality ci aiutano ad identificare i più importanti nodi della rete, evidenziando cosa rende un nodo importante.

### Degree Centrality

| Node | Degree Centrality |
|---|---|
| 6926 | 0.430084745763 |
| 6701 | 0.430084745763 |
| 6950 | 0.426906779661 |
| 7515 | 0.425847457627 |
| 7728 | 0.425847457627 |
| 7535 | 0.425847457627 |
| 7487 | 0.425847457627 |
| 6717 | 0.423728813559 |
| 7160 | 0.422669491525 |
| 6208 | 0.419491525424 |

Questa misura determina il numero di archi incidenti in un determinato nodo della rete, ovvero il numero di legami che un utente possiede.
Attraverso la Degree Distribution riusciamo ad individuare quei nodi con grado maggiore rispetto ad altri, che nei casi di reti naturali ci aiuterebbero ad evidenziare la presenza di hub, cosa che però nella nostra rete (come già visto in precedenza) risulta di difficile riuscita.
Nel grafico qui accanto, viene mostrata una classifica delle macchine linux (identificate dall'id unico) in ordine decrescente dal nodo con maggiore grado di centralità.

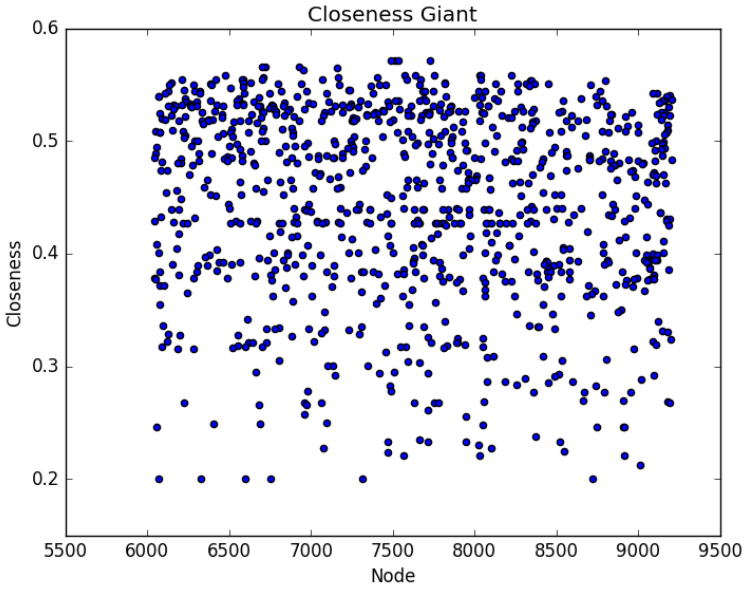
**Betweenness Centrality**

| Node | Betweenness |
|---|---|
| 9180 | 0.0189069680993 |
| 7628 | 0.0157840698997 |
| 9154 | 0.01363677335 |
| 8579 | 0.0107354163149 |
| 7254 | 0.0106122047134 |
| 9051 | 0.00820106561845 |
| 7937 | 0.00800968896628 |
| 7668 | 0.00747622559811 |
| 7094 | 0.00701522817549 |
| 6804 | 0.00693133196747 |

Questa misura ci darà indicazioni sulla centralità e influenza di un nodo all'interno della rete.
Per ogni nodo, tale misura viene definita come il numero totale di cammini minimi che lo attraversano.
Analiticamente sarà possibile calcolarla con la seguente formula:

$$g(v) = \sum_{s \neq t \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

dove $\sigma_{s,t}$ è il numero totale di cammini minimi da un nodo s ad un nodo t, mentre $\sigma_{s,t}$(v) è il numero totale di cammini minimi che passano attraverso il nodo v.
In pratica è una misura che calcola l'influenza di un nodo sul flusso di informazioni tra altri nodi, nel caso in cui il flusso segue un percorso minimo.
Anche in questo caso, viene riportato l'elenco dei primi nodi messi in ordine secondo il loro valore di betweennes.




**Closeness Centrality**

| Node | Closeness |
|---|---|
| 7515 | 0.571082879613 |
| 7728 | 0.571082879613 |
| 7535 | 0.571082879613 |
| 7487 | 0.571082879613 |
| 6717 | 0.565947242206 |
| 6926 | 0.565608148592 |
| 6701 | 0.565608148592 |
| 7160 | 0.564593301435 |
| 6950 | 0.562909958259 |
| 7638 | 0.557919621749 |

Questa misura esprime il grado in cui un nodo della rete è vicino a tutti gli altri, ovvero può essere rappresentato come l'inverso della somma degli Shortest Distance tra ogni nodo e ogni altro nodo nella rete.

## Random Network Comparison (Erdos Renyi Network)

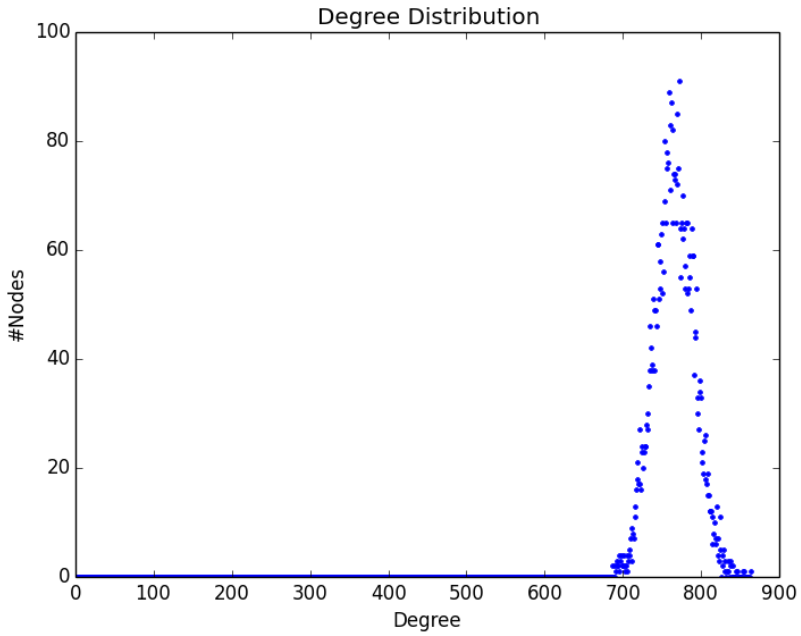
L'analisi è stata sviluppata in maniera analitica basandosi sul modello Random Erdos-Renyi Network, creando una rete attraverso la libreria NetworkX utilizzando come parametri il numero di nodi della nostra Linux Network e la probabilità p che un arco sia incluso nella rete indipendentemente degli altri.
Iniziamo definendo in primis l'*Average Degree,* calcolato come segue:

$$< k >= \frac{2L}{N} \ \ oppure \ \ p * (N-1) \ \approx 765$$

Da tale parametro potremo costruire la distribuzione, che per la Random Network si ricava con la seguente formula:

$$P(k) = \binom{N-1}{k} p^k (1-p)^{(N-1)-k}$$

dove rispettivamente avremo il numero di modi in cui possiamo selezionare k archi da N-1 potenziali archi che un nodo può avere, la probabilità di avere k archi, e la probabilità che i restanti (N-1-k) archi non ci siano.




Dal grafico della distribuzione è possibile osservare un addensamento intorno ad un grado di 1400 ed ha un andamento a campana. Seppur come già visto la *Degree Distribution* della Linux Network sia di difficile interpretazione, a livello generico la principale differenza fra una Random Network e una rete reale, è che nell'ultima c'è la presenza di hub ed un andamento convesso che si avvicina molto alla Power Law.
La distanza media tra le coppie di nodi può essere stimato mediante la seguente formula:

$$< d >= \frac{Log\ N}{Log < k >} = \frac{Log(4786)}{Log(765)} = \ 1.276145$$

Il *Clustering Coefficient* invece sarà possibile calcolarlo nel seguente modo:

$$C = \frac{< k >}{N} = \frac{765}{4786} = 0{,}159841$$

Risulterà molto più basso rispetto alla nostra rete, in quanto la porzione di vicini connessi tra loro risulta inferiore nella Random Network, come avviene tipicamente nelle reti naturali

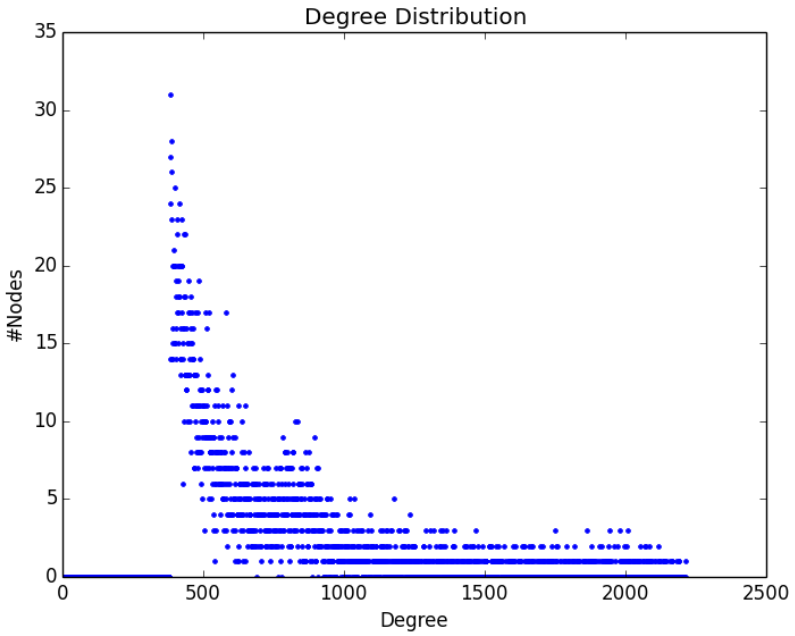
| ***Random Network*** | | ***Linux Network*** |
|---|---|---|
| *4786* | *N* | *4786* |
| *1830539* | *L* | *1829566* |
| *765* | *AVG Degree* | *765* |
| *0.159780*[3] | *Density* | *0.159780* |
| 0,159841 | *Clustering Coefficient* | *0.647806* |
| 1.276145 | *Average Path Lenght* | *2.330747* |

[3] Density di una Real Network equivale alla probabilità p nella Random Network

## Preferential Attachment Network (Barabasi Network)

Nel caso della Preferential Attachment Network è stata scelta la Barabasi Network, che attraverso la funzione predefinita presente su NetworkX con paramentri: (N) numero nodi e (m) numero archi.
La rete Barabasi inizialmente avrà m0 nodi e i link a questi nodi sono scelti arbitrarmene purché ogni nodo abbia almeno un collegamento, e si costruirà seguendo due fasi: Growth (ad ogni passo t si aggiunge un nodo collegandolo agli m nodi già presenti nella rete ) e Preferential Attachment (meccanismo probabilistico dipendente dal grado). Alla fine degli step temporali t , la rete di Barabasi avrà il numero di nodi pari a N = $m_0$ + t , e $m_0$ + m*t collegamenti.
La rete ottenuta avrà una Power-Law *Degree Distribution* con *Degree Exponent* γ=3

$$P(k) \sim k^{-3}$$




Il grado di ogni nodo si incrementa seguendo la stessa Power-Law con lo stesso Dynamic Exponent ß=1/2, e la crescita del grado è sub-linear, questo perchè ogni nuovo nodo ha molti più nodi da collegare rispetto ai precedenti nodi. Il primo nodo i aggiunto avrà il grado maggiore, quindi gli hubs sono grandi perchè essi arrivarono prima (fenomeno detto first-moved advantage).
La Degree Distribution segue una power-law con degree exponent γ = 3. Quest'ultimo caratterizza la topologia della rete, mentre il dynamical exponent β caratterizza l'evoluzione temporale del nodo. Questo rivela una profonda relazione tra la network's topology and dynamics.
Come nelle reti analizzate in precedenza, l'*Average Path Lenght* rappresenta la media distanza nella rete Barabàsi, per m>1

$$< d > = \frac{Log\ N}{Log\ Log\ N} = \frac{Log\ (4786)}{Log\ Log\ (4786)} = 3.965230227$$

Il diametro cresce più lento rispetto LogN, rendendo le distanze nel modello Barabási inferiori alle distanze osservate in una Random Network o Linux Network di dimensioni simili, con differenza particolarmente rilevante per grande N.
Il Clustering Coefficient è molto diverso da quello ottenuto in una Random Network e Linux Network:

$$C = \frac{(Log\ N)^2}{N} = \frac{(Log\ 4786)^2}{4786} = 0.0150019$$

La differenza sta nel termine $(Ln\ N)^2$ , che aumenta il coefficiente di clustering per grandi N, e di conseguenza la rete Barabási è localmente più clusterizzata.

| ***Barabasi Network*** | | ***Linux Network*** |
|---|---|---|
| *4786* | *N* | *4786* |
| *1682328* | *L* | *1829566* |
| *765* | *AVG Degree* | *765* |
| $H(1)\|_{m_0=1} = 4 - \pi$ | *Density* | *0.159780* |
| *0.0150019* | *Clustering Coefficient* | *0.647806* |
| *3.965230227* | *Average Path Lenght* | *2.330747* |

## Strumenti e dettagli implementativi

Il progetto si divide in una parte di creazione della rete ed una di analisi. Per ogni una sono utilizzati strumenti diversi, nello specifico:

- *Python* per la creazione e l'analisi
- *Gephi* per la visualizzazione

La creazione della rete si basa sui concetti spiegati precedentemente, quindi sul calcolo delle distanze tra ogni nodo. La complessità della funzione di distanza ricopre un ruolo centrale, in quanto si occupa di processare milioni di valori. Questa complessità è dovuta principalmente alle funzioni che calcolano la distanza tra ogni attributo, in particolare country_dist().
La prima calcola la distanza tra due location recuperandoli da una matrice contenente le distanze tra tutte le location presenti nel dataset, precedentemente salvata su file utilizzando l'API *ArcGIS* contenuta nel modulo python *geopy*[4].
I moduli python *pandas*[5] e *numpy*[6] forniscono le strutture dati adatte per la rappresentazione di grandi quantità di dati, grazie ai quali è stato possibile maneggiare file di grandissime dimensioni, affiancati a *matplotlib* per la creazione dei grafici. Per l'analisi della rete si utilizza il modulo *networkX*, contenente tutte le funzioni necessarie per tale scopo.
In fine si utilizza *Gephi* per la visualizzazione del grafo, producendo le immagini precedentemente illustrate.

L'intero progetto è disponibile al repository https://github.com/pigna90/ars-project.

[4] https://geopy.readthedocs.io/en/1.10.0/
[5] http://pandas.pydata.org/
[6] http://www.numpy.org/