

Sommario

Lo scopo di questo lavoro è fare un'analisi di rete usando un campione di antenne telefoniche cellulari, ipotizzando di voler costruire una ipotetica *mesh network* con esse. Abbiamo scelto di analizzare il sistema delle antenne comprese entro il Raccordo Anulare di Roma. Non sapendo a priori di quale tipo di rete si possa trattare, e che tipo di grafo si costruisce da essa, sono state studiate alcune proprietà topologiche e comportamentali dell'insieme di antenne studiate.

Dopo aver definito il criterio con cui dare un link a due nodi, sono state calcolate le matrici di adiacenza della rete complessiva e di quelle composte solo dalle antenne dei singoli gestori presenti in Italia. La prima parte dell'analisi consiste nell'estrapolare dai grafi ottenuti le distribuzioni dei gradi e nell'affidare all'istogramma dei gradi la forma funzionale che meglio gli si adatti. Avendo così ottenuto indicazioni significative sui tipi di rete in analisi, le abbiamo verificate studiando il comportamento percolativo della rete, mediante rimozione di nodi in due diversi scenari: un attacco intenzionale, che rimuovesse le antenne a partire da quelle con maggior grado, e una caduta random del sistema. I risultati ottenuti sono stati confrontati con modelli di rete esponenziali e scale-free. L'esposizione è articolata nel seguente modo:

Nella sezione 1 verrà fatta una panoramica sui moderni servizi di geolocalizzazione non basati sul GPS, usati da Google prima e ora in forma più aperta dalla Mozilla Foundation.

Nella sezione 2 verranno esposti i dati sulle antenne cellulari fornite dal Mozilla Location Service, e ne verrà analizzato in dettaglio il campione relativo alla città di Roma, entro il raccordo anulare.

Nella sezione 1 verrà fatta una panoramica sui moderni servizi di geolocalizzazione non basati sul GPS, usati da Google prima e ora in forma più aperta dalla Mozilla Foundation.

Nella sezione 3 si esporranno le basi della teoria delle reti e dei più importanti modelli di generazione, applicati allo studio della rete definita dalle antenne cellulari.

Nella sezione 4 si spiegherà come usare la teoria percolativa nello studio di una rete complessa già costruita. Si studierà quindi, piuttosto che la soglia critica di percolazione in costruzione, quale soglia si possa avere quando alla rete si tolgono nodi.

Nella sezione 5 viene effettuato lo studio percolativo mediante i due scenari di rimozione di nodi, analizzando in particolare l'andamento del diametro e delle dimensioni del cluster più grande con la progressiva caduta delle antenne.

Indice

| | |
|---|-----------|
| 1 Going beyond GPS localization | 4 |
| 1.1 Google Location Service | 4 |
| 1.2 Mozilla Location Service | 5 |
| 2 MLS dataset | 8 |
| 2.1 Data selection | 9 |
| 2.2 Analisi del raggio di copertura delle antenne | 12 |
| 3 Teoria delle reti | 14 |
| 3.1 Proprietà e grandezze caratteristiche | 14 |
| 3.2 Costruzione della rete e matrice di adiacenza | 15 |
| 3.3 Distribuzione del grado | 16 |
| 3.4 Modelli di rete | 17 |
| 4 Teoria della percolazione | 25 |
| 4.1 Soglia percolativa | 25 |
| 5 Network breakdown | 28 |
| 5.1 Speedup del codice (parte 1: multiprocessing) | 28 |
| 5.2 Strategie di attacco | 30 |
| 5.3 Speedup del codice (parte 2: graph-tool) | 31 |
| 5.4 Il punto di partenza | 32 |
| 5.5 Risultati | 33 |
| 6 Conclusioni | 36 |
| Riferimenti bibliografici | 37 |

Elenco delle figure

| | | |
|----|--|----|
| 1 | Logo del MLS | 5 |
| 2 | <i>Screen</i> dell'applicazione attiva | 6 |
| 3 | Immagine della mappa fornita da Mozilla | 6 |
| 4 | Scatterplot non georeferenziato delle antenne di Roma. | 11 |
| 5 | Heatmap delle antenne telefoniche di Roma | 12 |
| 6 | Istogramma log-log dei raggi con log-binning. | 12 |
| 7 | Frequency rank dei raggi in scala log-log | 13 |
| 8 | Distribuzione dei gradi per le 5 reti analizzate | 16 |
| 9 | Frequency rank dei gradi per le 5 reti analizzate | 17 |
| 10 | Reticolo ciclico. | 19 |
| 11 | Generazione con modello Watts-Strogatz. | 19 |
| 12 | Confronto grafi random. | 20 |
| 13 | Confronto clustering. | 21 |
| 14 | Albero scale-free | 22 |
| 15 | Confronto $P(k)$ | 24 |
| 16 | Esistenza soluzione non banale. | 26 |
| 17 | Confronto metodi. | 31 |
| 18 | Risultati random. | 34 |
| 19 | Risultati attacco. | 35 |

1 Going beyond GPS localization

1.1 Google Location Service

GPS and battery drain

La geolocalizzazione dei dispositivi, soprattutto quelli mobile, sta rapidamente diventando parte intergrante del nostro modo quotidiano di fruire la tecnologia. Si usa nel navigatore stradale, per impostare automaticamente il luogo di cui desideriamo le previsioni del tempo o da cui vogliamo prendere un treno o un aereo, per aggiungere informazioni contestuali alle fotografie che scattiamo e tanto altro ancora.

Il modo storicamente utilizzato per effettuare la geolocalizzazione è per via satellitare, tramite il GPS. Ma il GPS ha due tipi di problemi:

- è *davvero* lento ad agganciare un numero congruo di satelliti, tali da poter registrare una posizione accurata,
- consuma molta energia.

Soprattutto il secondo punto è in palese conflitto con l'esigenza dei moderni dispositivi portabili di avere una ampia autonomia energetica.

Per porre rimedio a questo fatto, Google ha via via cominciato ad utilizzare altri tipi alternativi di geolocalizzazione, per comporre un sistema ibrido.

1. Un primo grado di grossolana geolocalizzazione avviene tramite le celle della rete telefonica, a cui tutti gli smartphone sono connessi. La precisione è dell'ordine delle centinaia di metri.
2. Un secondo grado, molto più accurato, viene realizzato tramite le reti wifi visibili in quell'istante dal dispositivo. La precisione è dell'ordine di qualche decina di metri.
3. Infine, solo se il compito richiesto richiede una precisione ulteriore, si accende il GPS, per un risultato con la precisione inferiore al metro. Il cominciare da una stima della posizione più che accettabile riduce di molto la durata delle operazioni a GPS acceso, consentendo un drastico risparmio della batteria.

Geolocation via WiFi

La diffusione del Wifi è ormai capillare, soprattutto in contesto cittadino. Virtualmente c'è un router WiFi in ogni appartamento. Mentre camminiamo per la città col WiFi dello smartphone acceso, captiamo in ogni punto decine di segnali. Se è nota la posizione di questi router e la potenza del segnale da loro emesso è possibile triangolare la nostra posizione con una notevole precisione, dovuta sia al fatto che un segnale WiFi ha un raggio tipico di una trentina di metri, sia dovuto all'ingente numero di segnali coi quali si sta triangolando, tipicamente più di una decina.

Quindi per sapere la mia posizione devo avere accesso a una mappa delle posizioni di tutti i router. E come si costruisce questa mappa? Esattamente al contrario: camminando per la città con GPS acceso, avendo dunque un'alta precisione sulla posizione

del dispositivo, e triangolando la posizione di tutti i router visibili combinando tutte le rilevazioni del tracciato.

Dunque Google ha mappato tutti i WiFi di tutto il globo per dare la possibilità agli utenti dei suoi servizi come Google Maps (e delle applicazioni che si appoggiano alle sue API) di ottenere la propria localizzazione con una buona precisione è un consumo di batteria risibile. La precisione è arrivata ad essere dell'ordine dei 5 metri e i tempi di accesso pressoché istantanei: di fatto rendendo inutile accendere il GPS nella maggior parte dei casi.

Questo tipo di localizzazione ha inoltre un altro grande vantaggio: funziona anche sottoterra e dentro gli edifici, luoghi normalmente inaccessibili al segnale GPS, di natura satellitare.

Ovviamente la mappa dei router WiFi è in continua evoluzione, per cui il modo più conveniente per redigerla è sfruttando il crowdsourcing: ai milioni di utenti ignari giornalmente in moto per la città viene acceso il GPS un paio di volte al giorno per pochi secondi, in un momento di inutilizzo del dispositivo, e in questo modo si crea velocemente una mappa complessiva e quotidianamente aggiornata, a beneficio di tutti.

Tutto questo è tremendamente intelligente ed efficiente. C'è un solo grande problema: i dati sono chiusi.

1.2 Mozilla Location Service



Figura 1: Logo del MLS

Mozilla è da sempre impegnata nello sviluppo di tecnologie web aperte e standardizzate. Avendo riconosciuta la centralità della geolocalizzazione e la crescita esponenziale di siti ed applicazioni location-aware, ha deciso di ricalcare le orme di Google e fondare il suo servizio di geolocalizzazione usando WiFi ed antenne cellulari: *Mozilla Location Service*.

L'obbiettivo è quello di geolocalizzare gli utenti sulla base dell'ambiente radio che li circonda: MLS è un progetto collaborativo per creare un database mondiale aperto di Cell ID e WiFi georeferenziati. La mappatura viene fatta dagli utenti su base puramente volontaria, utilizzando l'apposita applicazione *Mozilla Stumbler*.

Il progetto è ormai maturo e, come si può osservare nelle mappe in fig. 2 e 3 la copertura del mondo è molto capillare.

- La prima mostra un esempio di utilizzo del programma a San Lorenzo, nel percorso tra le nostre due abitazioni. I pallini verdi sono i punti GPS delle rilevazioni complete (WiFi + rete cellulare) effettuate durante il tragitto. Come si può notare sono piuttosto fitti: 4-5 per ogni isolato. Le aree sfumate in blu mostrano invece, in maniera approssimata per motivi di privacy, l'esito dell'elaborazione di tutte le precedenti osservazioni nell'area, ovvero tutti i router WiFi e le antenne ricostruiti finora. La mappa è fittissima, ma questo non deve trarre in inganno: è ancora incompleta e per certi versi inaffidabile. Un esempio diretto sono le numerose ricostruzioni che appaiono in mezzo ai binari della stazione Termini: quelle derivano dai WiFi all'interno dei treni a cui si collegano i viaggiatori, ma che essendo

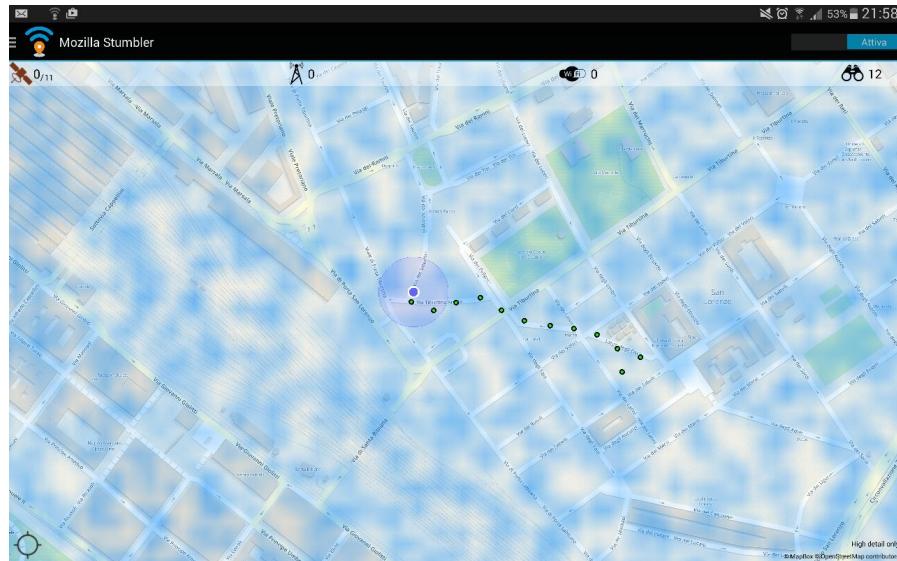


Figura 2: Screen dell'applicazione attiva

oggetti in moto non possono essere usati ai fini della geolocalizzazione e pertanto dovrebbero essere filtrati ed esclusi dal database.

- La seconda mappa mostra invece la copertura globale raggiunta dal progetto. Sebbene risulti uno stadio abbastanza avanzato, si può notare come ci siano ancora delle disparità di mappatura anche tra gli stati ad alta presenza tecnologica: si confrontino ad esempio le capitali della Cina e del Giappone. Inoltre, essendo un progetto collaborativo open source e con sede negli Stati Uniti, si può vedere come gli stati in cui la cultura open è meno diffusa o quelli in tensione con gli USA per quanto riguarda la politica estera risultino globalmente meno mappati. Un esempio emblematico potrebbe essere la differenza tra Nord Korea e Sud Korea.

Dato l'approccio di *crowdsourcing* collaborativo, i dati sono tanti e la copertura

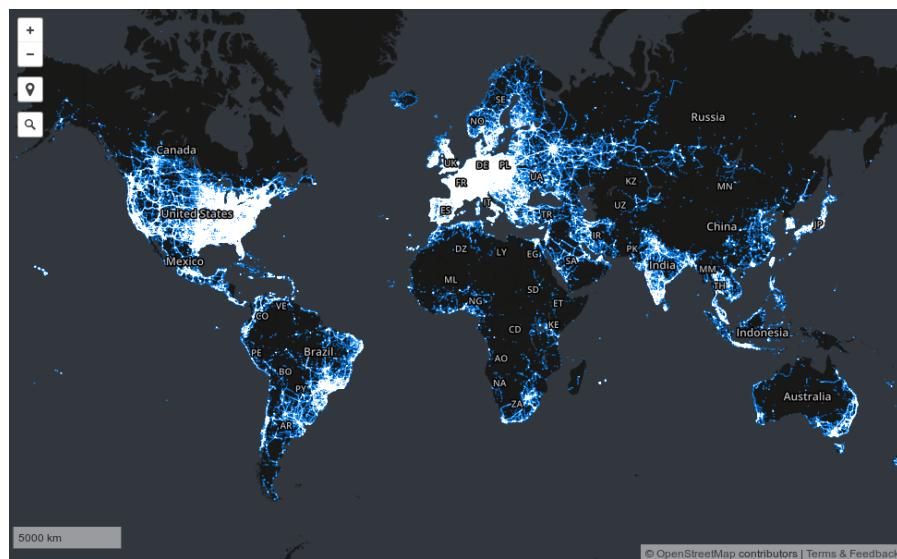


Figura 3: Immagine della mappa fornita da Mozilla

Tabella 1: Alcune statistiche dei dati forniti da MLS

| | data | total | |
|-----------------------|---------|----------------|---------------|
| Wifi Networks | | 521.610.000 | |
| Wifi Observations | | 1.1006.560.000 | |
| MLS Cells | | 20.390.000 | |
| MLS Cell Observations | | 2.468.460.000 | |
| OpenCellID Cells | | 7.990.000 | |
| | network | Italy | United States |
| GSM (2G) | | 132.468 | 230.236 |
| UMTS (3G) | | 516.379 | 976.603 |
| LTE (4G) | | 96.489 | 1.595.402 |
| Total Cells | | 745.336 | 2.802.241 |
| WiFis | | 9.844.682 | 175.599.375 |

mondiale, continuamente aggiornati, aperti, facilmente scaricabili in database ordinati, puliti e ben documentati. Di seguito sono elencate alcune statistiche per avere un'idea dei numeri in gioco:

Mozilla Location Service funziona sia con le antenne cellulari che con i WiFi, soltanto che attualmente i dati della mappa WiFi (mostrati approssimati nella mappa precedente) non possono essere resi pubblici per questioni relative alle norme sulla privacy vigenti sia negli Stati Uniti sia in altre nazioni.

Pertanto gli unici dati attualmente liberamente disponibili (dominio pubblico, licenza CC0) sono quelli sulle antenne radio delle varie generazioni: 2G (GSM), 3G (UMTS), 4G (LTE). I dati forniti da Mozilla sono un'estensione di quelli già disponibili sulla piattaforma *OpenCellID*, anche loro open (licenza CC-BY-SA 3.0).

Questi sono dunque i dati che abbiamo analizzato in questa nostra tesina.

2 MLS dataset

I dati scaricabili dal sito del progetto Mozilla Location Service appaiono come in tabella 2

I dati vengono forniti in un file CSV, le cui voci, secondo la documentazione ufficiale, sono organizzate nella seguente maniera:

radio Network type. One of the strings GSM, UMTS or LTE.

mcc Mobile Country Code. An integer, for example 505, the code for Australia.

net For GSM, UMTS and LTE networks, this is the mobile network code (MNC). An integer, for example 4, the MNC used by Vodafone in the Netherlands.

area For GSM and UMTS networks, this is the location area code (LAC). For LTE networks, this is the tracking area code (TAC). An integer, for example 2035.

cell For GSM and LTE networks, this is the cell id or cell identity (CID). For UMTS networks this is the UTRAN cell id, which is the concatenation of 2 bytes of radio network controller (RNC) code and 2 bytes of cell id. An integer, for example 32345.

unit For UMTS networks, this is the primary scrambling code (PSC). For LTE networks, this is the physical cell id (PCI). For GSM networks, this is empty. An integer, for example 312.

lon Longitude in degrees between -180.0 and 180.0 using the WSG 84 reference system. A floating point number, for example 52.3456789.

lat Latitude in degrees between -90.0 and 90.0 using the WSG 84 reference system. A floating point number, for example -10.034.

range Estimate of radio range, in meters. This is an estimate on how large each cell area is, as a radius around the estimated position and is based on the observations or a knowledgeable source. An integer, for example 2500.

samples Total number of observations used to calculate the estimated position, range and averageSignal. An integer, for example 1200.

changeable Whether or not this cell is a position estimate based on observations, and therefore subject to change in the future, or is an exact location entered from a knowledgeable source. A boolean value, encoded as either 1 (for “changeable”) or 0 (for “exact”).

created Timestamp of the time when this record was first created. An integer, counting seconds since the UTC Unix Epoch of 1970-01-01T00:00:00Z. For example, 1406204196, which is the timestamp for 2014-07-24T12:16:36Z.

updated Timestamp of the time when this record was most recently modified. An integer, counting seconds since the UTC Unix Epoch of 1970-01-01T00:00:00Z. For example, 1406204196, which is the timestamp for 2014-07-24T12:16:36Z.

averageSignal Average signal strength from all observations for the cell network. An integer value, in dBm. For example, -72.

This field is only used by the OpenCellID project and historically has been used as a hint towards the quality of the position estimate.

I dati forniti sono di tipo aggregato, ovvero riportano il numero di osservazioni per antenna ed il risultato della stima della sua posizione. Esistono anche dati grezzi, ovvero corrispondenti alle singole osservazioni dei singoli utenti. Tale dataset però non viene reso pubblicamente disponibile, in quanto contenente diverse informazioni, anche di carattere dinamico, utili a tracciare il singolo individuo. È allo studio un meccanismo di autorizzazioni per permettere agli utenti consci degli eventuali rischi di pubblicare i dati che li riguardano.

I valori di latitudine e longitudine per le coordinate nelle proiezioni geografiche seguono il sistema di riferimento dettato dalla convenzione WSG 84 Web Mercator.

Tabella 2: Esempio del dataframe MLS

| radio | mcc | net | area | cell |
|------------|------------|------------|---------------|---------|
| GSM | 262 | 7 | 20205 | 2227 |
| GSM | 262 | 2 | 685 | 6132 |
| GSM | 262 | 1 | 14660 | 39399 |
| GSM | 262 | 7 | 20202 | 33063 |
| GSM | 262 | 7 | 20205 | 2472 |
| unit | lon | lat | range | samples |
| NaN | 13.308816 | 52.512122 | 147481 | 14586 |
| NaN | 7.018766 | 49.223543 | 14976 | 5336 |
| NaN | 9.879624 | 51.335388 | 101430 | 4049 |
| NaN | 13.327494 | 52.516999 | 148854 | 6412 |
| NaN | 13.296922 | 52.507817 | 132401 | 11216 |
| changeable | created | updated | averageSignal | |
| 1 | 1299573448 | 1453335612 | -64 | |
| 1 | 1299584114 | 1453337312 | -74 | |
| 1 | 1299693112 | 1453335486 | -57 | |
| 1 | 1301668631 | 1453335612 | -61 | |
| 1 | 1301668631 | 1453335612 | -65 | |

2.1 Data selection

Una volta scaricati i dati aggregati dalla pagina web <https://location.services.mozilla.com/downloads> sono cominciate le operazioni di selezione dei dati. Il data sample riguarda tutti i continenti e risulta molto grosso (un file csv di circa 650 MB), per cui è necessario ridurlo il più possibile per poterlo maneggiare col nostro limitato quantitativo di RAM.

Una prima grossolana ma efficiente scrematura riguarda i dati caratterizzati da un mobile county code non italiano, si è pertanto imposta la condizione

```
mcc == 222
```

Successivamente vengono scartati i dati ritenuti inaffidabili, ovvero con soltanto una rivelazione da parte degli utenti

```
samples > 1
```

Adesso che il datasample si è ridotto molto possiamo effettuare delle operazioni computazionalmente un po' più pesanti: vogliamo eliminare tutte le rilevazioni al di fuori del Grande Raccordo Anulare, che per semplicità è stato schematizzato come una circonferenza di raggio 10 km con centro esattamente nel Colosseo. Per far questo serve definire una nozione di distanza. Dato che i nostri sono dati geolocalizzati sarebbe naturale introdurre una distanza geodesica. A tal fine abbiamo usato la libreria *geopy*, che contiene due definizioni differenti:

- Great circle distance: la distanza geodesica su una sfera. Per due punti non agli antipodi passa sempre una circonferenza di raggio massimo lungo cui scorre la geodesica, ovvero il cammino di minima distanza.
- Vincenty distance: la distanza geodesica su un ellissoide oblato. Tale distanza tiene in conto che la Terra non è una sfera perfetta ma è invece leggermente schiacciata ai poli. Delle due è quella più accurata, ma anche quella più difficile da calcolare numericamente.

L'ulteriore condizione da soddisfare per i dati risulta dunque

```
geodesicDistance(place) <= raggioRaccordoAnulare
```

Si sono fatte differenti prove sia con *vincenty* (più lenta) che con *great_circle* (leggermente più veloce), ma i tempi di calcolo risultavano comunque spropositati. Pertanto abbiamo fatto una approssimazione: dato che la città di Roma sottende un angolo solido minuscolo rispetto alla totalità del pianeta, abbiamo ritenuto accettabile usare una distanza euclidea, ovviamente trasformando in metri le coordinate angolari di latitudine e longitudine con i relativi fattori di scala, dettati dal raggio terrestre.

La distanza euclidea coinvolge solo quadrati è radici quadrate, risultando nel complesso circa dieci volte più veloce delle altre due concorrenti. Il prezzo da pagare è una leggerissima imprecisione, del tutto trascurabile alle nostre scale. La funzione utilizzata è pertanto

```
def euclideanDistance(x,y):
    return numpy.sqrt(numpy.square(x) + numpy.square(y))
```

Da notare il fatto che per il calcolo algebrico è stata utilizzata la libreria *numpy* invece che la libreria *math* built-in in Python, poiché è più veloce (è scritta in C) e supporta le operazioni direttamente su vettori di coordinate.

I dati sono stati importati in un dataframe tabulare usando la libreria *pandas*. Questo che ci ha permesso di effettuare facilmente tutte le *query* necessarie per il filtraggio dei dati.

Tabella 3: Esempio del dataframe da noi utilizzato

| radio | net | cell | lon | lat | range | samples | distance | degree |
|-------|-----|-------|-----------|-----------|-------|---------|----------|--------|
| GSM | 10 | 29369 | 12.630353 | 41.892274 | 10568 | 2 | 11457 | 2412 |
| GSM | 10 | 28691 | 12.620040 | 41.890106 | 9674 | 2 | 10598 | 2361 |
| GSM | 88 | 19451 | 12.531354 | 41.986043 | 2172 | 4 | 11129 | 179 |
| GSM | 88 | 16608 | 12.605920 | 41.918988 | 4129 | 7 | 9953 | 621 |
| GSM | 88 | 55647 | 12.604219 | 41.928254 | 9245 | 23 | 10201 | 2048 |

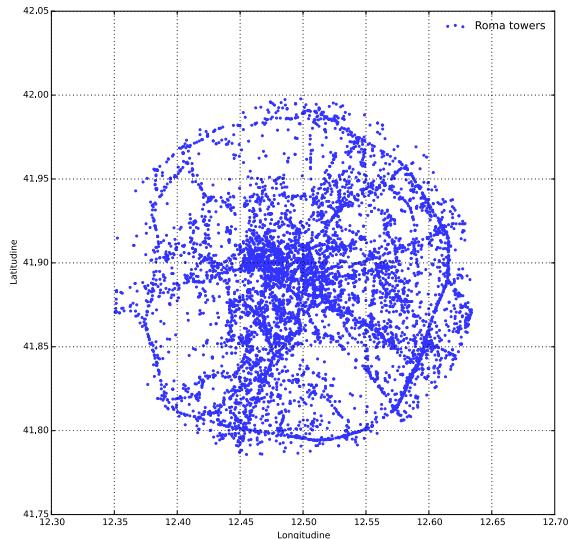
A questo punto abbiamo finalmente il nostro datasample della città di Roma: circa 7000 antenne in un file csv agevolmente maneggiabile di circa 1MB.

Per visualizzare agevolmente i nostri dati serve una mappa georeferenziata, preferibilmente interattiva. A tal fine per il notebook di IPython abbiamo usato la libreria *gmaps*, che dà semplice accesso inline alle mappe di Google Maps dando la possibilità di creare una *heatmap*, mentre per l'HTML di questa presentazione abbiamo usato le analoghe funzioni della libreria *gmplot*:

```
roma = pandas.read_csv("../data/Roma_towers.csv")
coordinate = roma[['lat', 'lon']].values
heatmap = gmaps.heatmap(coordinate)
gmaps.display(heatmap)

colosseo = (41.890183, 12.492369)
mappa = gmplot.GoogleMapPlotter(41.890183, 12.492369, 12)
mappa.heatmap(roma.lat.values,roma.lon.values)
mappa.draw("../doc/heatmap.html")
```

(per scrivere queste poche linee di codice c'è voluto un intero pomeriggio!)

**Figura 4:** Scatterplot non georeferenziato delle antenne di Roma.

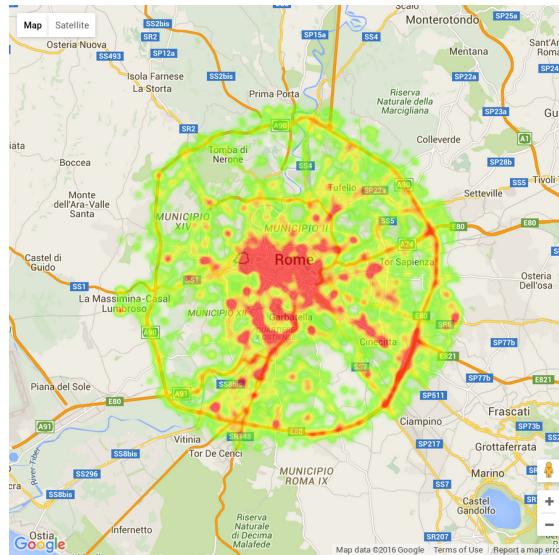


Figura 5: Heatmap delle antenne telefoniche di Roma

2.2 Analisi del raggio di copertura delle antenne

Dato che ci servirà fare grafici con scale logaritmiche eliminiamo i dati di antenne che presentano un raggio nullo

```
range =! 0
```

Il raggio minimo risulta essere 1m, mentre quello massimo 20341m. Dato che il raggio del Grande Raccordo Anulare è circa 10km questo significa che ci saranno antenne con un grado di connessione totale. Facciamo un istogramma log-log per la distribuzione del raggio di copertura, sia con la canalizzazione lineare sugli interi, sia con una canalizzazione logaritmica in base 2, per ridurre il rumore sulla coda.

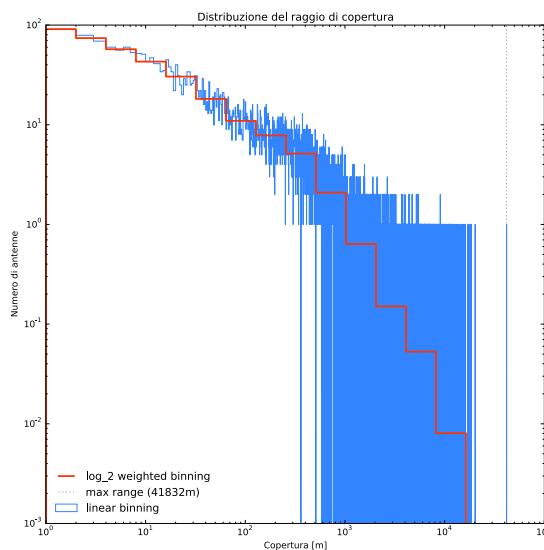


Figura 6: Istogramma log-log dei raggi con log-binning.

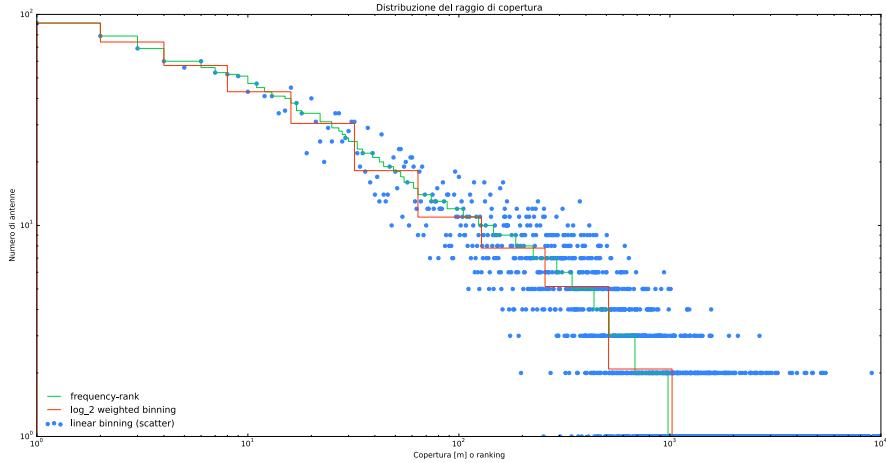


Figura 7: Frequency rank dei raggi in scala log-log

La canalizzazione logaritmica pesata permette di osservare l’andamento ben sotto il singolo conteggio, ampliando di una decade l’intervallo di osservazione.

In figura 6 si può vedere come l’andamento sia abbastanza power-law su diverse decadi, soprattutto fino a conteggio = 1. Per verificare ulteriormente questo fatto abbiamo generato anche la curva del frequency-rank (Figura 7), che risulta seguire senza esitazioni il trend delineato dagli istogrammi. Il frequency-rank si ottiene ordinando in maniera decrescente il numero di conteggi per ogni canale unitario e associando un relativo ranking intero decrescente ai raggi corrispondenti.

3 Teoria delle reti

Le reti sono insiemi di oggetti per i quali è possibile una connessione. Gli ultimi 60 anni hanno visto un crescente interesse per esse, perché con le reti è possibile descrivere sistemi dei più disparati tipi nei campi più vari, dalla biologia all'economia, passando per reti elettriche, informatiche, ecosistemi, e altro ancora. Ancora più importante, negli ultimi vent'anni è stato scoperto che una classe di reti ha proprietà del tutto comuni nonostante l'intrinseca diversità tra sistemi; la modellizzazione di queste reti, come cresciuta con caratteristiche preferenziali, è dovuta a Barabasi e Albert (1999), e vengono dette scale-free.

Per maggior semplicità, nell'esposizione del lavoro svolto verrà sempre assunto che le reti siano dirette e non pesate.

3.1 Proprietà e grandezze caratteristiche

Dal punto di vista matematico una rete è rappresentata da un grafo. Gli oggetti costitutivi di un grafo sono i **nodi**, i quali vengono collegati opportunamente tra loro con dei **link** secondo dei criteri che dipendono dal modello (nella costruzione di un grafo teorico) o dalla natura del sistema (per le reti reali). Il numero di nodi della rete, o di una sua parte, è ovviamente la grandezza fondamentale per definirne le dimensioni; il numero e la distribuzione dei link ne descrivono la connettività.

Prima di procedere alla descrizione dei modelli di rete sopracitati, è utile elencare un glossario di caratteristiche delle reti, spesso determinanti per distinguere un grafo scale-free dagli altri nello studio delle reti reali.

Grado Il grado di un nodo è il numero di altri nodi a cui è connesso. La connettività di una rete è ben rappresentata dalla **distribuzione del grado**, la cui forma funzionale $P(k)$ la cui forma funzionale è il primo criterio per discriminare una rete scale-free.

Distanze alla base di ogni topologia c'è il concetto di distanza. La distanza tra due nodi di un grafo è definita come il numero di link che li separa nel più piccolo cammino possibile lungo la rete. Altre quantità topologiche importanti sono:

- l'**eccentricità di un nodo**, è la massima delle distanze tra un nodo scelto e tutti gli altri nodi della rete;
- il **diametro** è la massima eccentricità tra quelle di tutti i nodi del grafo; detto in termini più generali, il diametro è il minor cammino più grande tra tutte le possibili coppie di nodi della rete;
- **average path length**, cioè la distanza media tra tutte le possibili coppie di nodi della rete.

Custering Per misurare quanto i nodi di un grafo tendono a creare dei clusters, viene definito il coefficiente di clustering C_i . Dal punto di vista di un vertice i di grado k_i , quindi, considerando i nodi a esso collegato, C_i è definito come il rapporto

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

dove E_i è il numero di link tra i k_i primi vicini del vertice e $k_i(k_i - 1)/2$ il numero di link possibili tra essi: cioè il numero di collegamenti necessari perché i con i suoi vicini sia una porzione di grafo *completa*, detta anche *clique*. La media su tutti i nodi dei rispettivi C_i dà un'indicazione su quanto la rete sia complessivamente clusterizzata e quindi può essere preso come coefficiente di clustering globale C della rete. Una definizione più recente di C , equivalente alla precedente, lo pone uguale al rapporto tra il numero di triplette di nodi completamente collegate N_Δ e il numero di triplette che vede i tre nodi collegati da almeno due link N_\wedge . In entrambi i casi, più C è vicino a 1, più si ha clusterizzazione.

Il coefficiente di clustering svolge un ruolo importante nel distinguere una rete completamente random da una che presenta caratteristiche di *small-world*: infatti un rete con piccolo diametro tende a avere un $\langle C \rangle$ maggiore di una simile rete puramente casuale costruita con lo stesso numero di nodi e con il medesimo cammino più corto medio. Questo comportamento è stato notato in molte reti reali (Watts e Strogatz 1998).

Centralità Esistono vari modi per definire quando un nodo è centrale rispetto ad altri.

Il primo è più immediato è il suo grado. Altri due tra i più importanti sono la **betweenness** e il **page-rank**. Preso un nodo i , il primo è definito come il numero di cammini più corti che è possibile tracciare tra due qualsiasi nodi della rete, purché passino per i ; il secondo dà una maggior centralità a i maggiore è il numero di link *diretti* verso di esso, tenendo conto anche del rank dei nodi che si collegano a esso.

In reti dirette, benché non siano *matematicamente* uguali, betweenness e page-rank sono statisticamente molto correlati al grado, pertanto non verranno analizzati.

3.2 Costruzione della rete e matrice di adiacenza

Prese le antenne nell'area selezionata come i nodi della rete, è necessario definire un criterio per stabilire un link. Nell'ottica di una ipotetica mesh network, la scelta è stata di considerare collegate due antenne se non hanno zone d'ombra del segnale nello spazio tra di esse, in modo che qualsiasi utente al suo interno possa comunicare con altri utenti (si suppone quindi un diverso metodo di comunicazione tra antenne, come cavo, altri canali Wi-fi). Trascurando dislivelli del terreno e edifici, è stato supposto che la porzione di spazio coperta da un'antenna sia un cilindro del raggio fornito dai dati. Il criterio scelto è stato quindi che la distanza δ_{ij} tra due antenne fosse minore dei loro raggi in modo tale che essi si sovrappongano per il 20%: $\delta_{ij} < 0.8(r_i + r_j)$.

Stabilito il criterio, bisogna costruire la rete, popolando la matrice di adiacenza dei nodi. Dato che il criterio di linking richiede la conoscenza della distanza tra due antenne, è stato necessario calcolare le distanze tra ogni coppia di nodi. I dati in nostro possesso forniscono le coordinate geografiche, che sono delle coordinate sferiche, e quindi ci si è posti il problema di come calcolare le distanze. Inizialmente la scelta è caduta su una funzione della libreria geopy, la quale permette di calcolare la distanza geodesica con il metodo great-circle e con la formula vincenty. La prima è più veloce e usa un modello sferico di Terra, la seconda richiede più calcoli perché usa un accurato modello

ellissoidale (impiega circa il doppio del tempo del metodo great-circle). Purtroppo, tuttavia, i tempi di calcolo richiesti da queste funzioni sono piuttosto alti in prospettiva di calcolare una matrice di 7000^2 elementi.

Per risparmiare tempo, dato che la rete definita non è diretta, per prima cosa è stata modificata la funzione di calcolo della matrice di adiacenza in modo che calcolasse soltanto la metà superiore. Successivamente si poteva tentare di diagonalizzare a blocchi la matrice, cosa purtroppo impossibile poiché dai dati risulta che un certo numero di antenne hanno raggi d'azione nell'ordine dei 10-20 km, ricoprendo quindi tutta l'area di Roma selezionata. L'unica via praticabile per far calare drasticamente il tempo necessario per calcolare tutte le matrici necessarie per l'analisi è stata quindi convertire le coordinate geografiche in cartesiane e usare la distanza euclidea. In questo modo si è ottenuto un guadagno di velocità di un fattore 10. Calcolate le matrici definitive per la rete complessiva e per quelle delle singole compagnie, sono state salvate su file in modo da non doverle più ricalcolare.

3.3 Distribuzione del grado

I vettore di gradi per ogni nodo del grafo si ottiene con il metodo `.degree()` della classe Graph di networkx. Abbiamo dunque fatto un istogramma congiunto di tutti i provider e della rete totale con un canale per ogni possibile numero naturale, ma in scala log-log risultava illegibile. Per ridurre un poco il rumore sulle code la canalizzazione è stata ridotta di 8 volte rispetto alle unità naturali. Data la poca estensione delle curve ed il fatto che molte non arrivavano fino ad 1, si è scelto di evitare il log-binning in base 2, in quanto in questo caso troppo grossolano e non in grado di conservare tutta l'informazione sulle code della distribuzione.

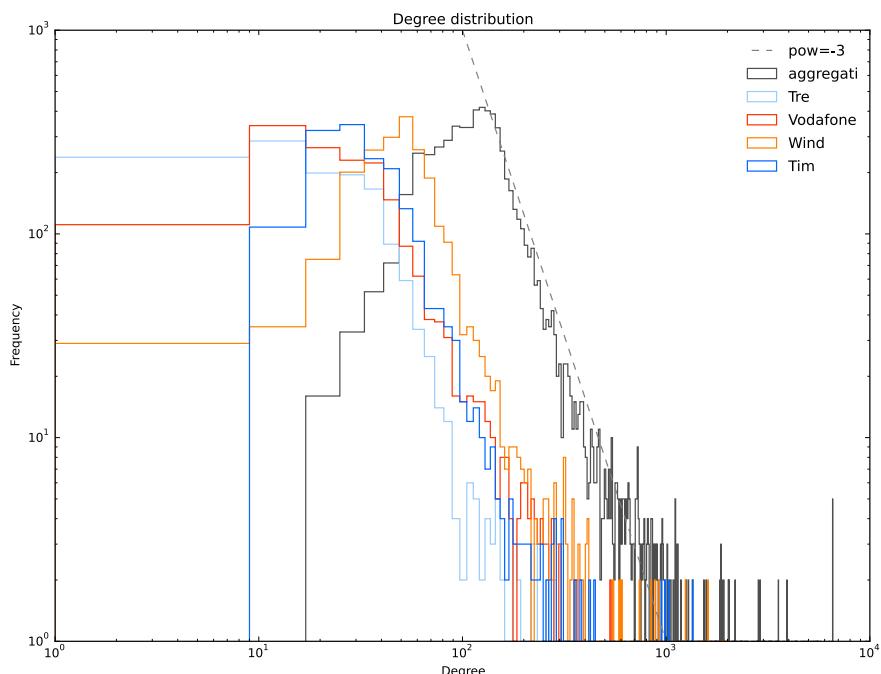


Figura 8: Distribuzione dei gradi per le 5 reti analizzate

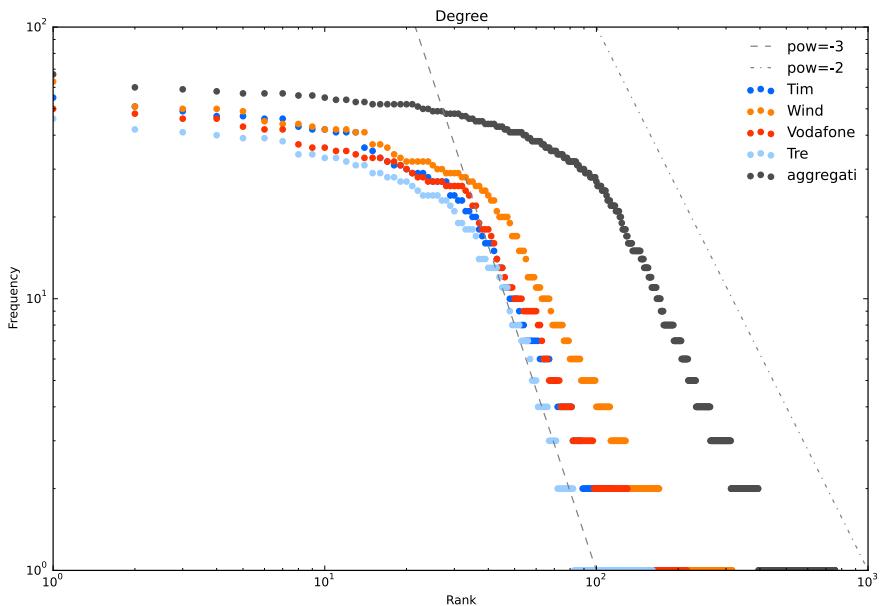


Figura 9: Frequency rank dei gradi per le 5 reti analizzate

Come si può vedere nella figura 8, soprattutto per la rete complessiva l’andamento è circa power law dal massimo in poi. L’esponente è in valore assoluto di poco superiore a 3 nella prima parte e di poco inferiore a 3 nell’ultima parte della coda, estesa per circa mezza decade.

In figura 9 viene anche riportato l’andamento del frequency-rank (in forma di scatterplot per migliorarne la legibilità). La differenza nei valori sulle y rispetto all’istogramma precedente è dovuta al fatto che grazie all’ottima visibilità qui è stato possibile riprendere la canalizzazione sui numeri naturali, non aggregata.

3.4 Modelli di rete

Costruite le reti e individuate le distribuzioni del grado che le caratterizzano, è utile avere alcuni importanti modelli di rete come paragone.

Lo studio delle reti random nasce dagli ungheresi Erdős e Rényi (1959), i quali per primi modellizzarono la definizione di una rete usando criteri probabilistici. Nel corso dei 40 anni successivi è stato osservato che molte reti reali, a dispetto delle dimensioni, avessero un diametro molto piccolo, portando alla definizione del concetto di *small-worlds* e al modello di Watts e Strogatz (1998).

In entrambi i casi si parte da una configurazione iniziale di nodi per poi randomizzare i link tra essi. Per questo motivo la distribuzione del grado segue una Poissoniana, con una coda destra caratteristica di forma esponenziale; con un numero sufficiente di nodi e link, la distribuzione del grado si approssima a una gaussiana con un grado medio ben definito.

3.4.1 Random network

Il modello di Erdős e Rényi parte da un certo numero N di nodi e n di link. Se poniamo che i nodi siano distinguibili, possono esistere $\binom{n}{N(N-1)/2}$ configurazioni equiprobabili.

bili tra le quali può esserne presa una in maniera random. Una maniera più articolata per definire una rete random è il criterio binomiale: partire da un set di N nodi e dare a ogni possibile coppia un link con una certa probabilità p . Il punto più importante di un approccio probabilistico alle reti è lo studio di proprietà dei grafi: ponendo $N \rightarrow \infty$, se la probabilità che una certa proprietà si verifichi tende a 1, si osserva che per reti limitate quella proprietà si verifica con probabilità significativa anche con pochi nodi. Ciò si verifica per molte proprietà, e questo fatto permette di poter trattare le reti per categorie secondo le loro proprietà peculiari.

Nel caso di un grafo di Erdős e Renyi queste sono:

- la distribuzione del grado ha una forma di distribuzione binomiale, la quale tende a una poissoniana per p piccole, e a una gaussiana per $\langle k \rangle$ grandi. Questo implica che la topologia della rete è abbastanza omogenea, con molti nodi che hanno approssimativamente lo stesso grado;
- con N grande, il diametro tende a essere piccolo, come l'average path length. Con p non troppo piccolo il numero di nodi che abbiano una certa distanza una lunghezza l si può approssimare a $\langle k \rangle^l$; uguagliandolo a N deriva che diametro e average path length scalano con buona approssimazione con il logaritmo di N , secondo la relazione

$$l \sim \frac{\ln(N)}{\ln(\langle k \rangle)}. \quad (1)$$

Molte reti reali presentano simili caratteristiche nei gradi di separazione, che hanno portato alla definizione del termine "small world" per esse;

- il clustering di una rete random tende a essere molto basso. Infatti, preso un nodo e i suoi primi vicini, la probabilità una coppia di essi sia connessa è p . Pertanto su tutto il grafo, il coefficiente di clustering medio è proprio p , il quale di solito è abbastanza minore di 1 (con $p = 0.1$ un grafo random diventa già molto connesso, per esempio con 10^4 nodi avrebbe $\langle k \rangle = 10^3$). Questo fatto, al contrario del diametro, si pone in contrasto con le reti reali, le quali hanno quasi sempre un $\langle C \rangle$ sensibilmente più alto.

3.4.2 Small World

Come abbiamo visto il modello di Erdős e Renyi descrive bene il piccolo diametro delle reti reali, ma non il loro elevato grado di clusterizzazione. Inoltre il coefficiente di clustering di esse è simile per reti con numero di nodi molto diverso. Notando per primi ciò, Watts e Strogatz hanno formulato un modello che meglio si adattasse alle caratteristiche reali delle reti.

Il fatto che il coefficiente di clustering non dipende dal numero di nodi è caratteristico dei reticolati, pertanto il punto di partenza del modello di Watts e Strogatz è un reticolo con condizioni al contorno cicliche, i cui N nodi sono collegati ai primi n vicini. Se poniamo, per esempio, $n = 2$, si configura così un anello come in Figura 10.

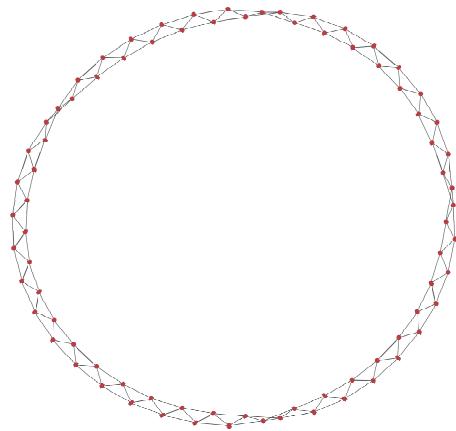


Figura 10: Il punto di partenza del modello Watts-Strogatz: un reticolo con condizioni al contorno cicliche.

Successivamente si procede a riarrangiare in maniera random i link tra i nodi, con una probabilità p per ogni link di venire modificato. In questo modo si hanno un certo numero di link (in media pN) che invece di essere tra nodi in prossimità, saranno tra nodi più lontani, come in Figura 11.

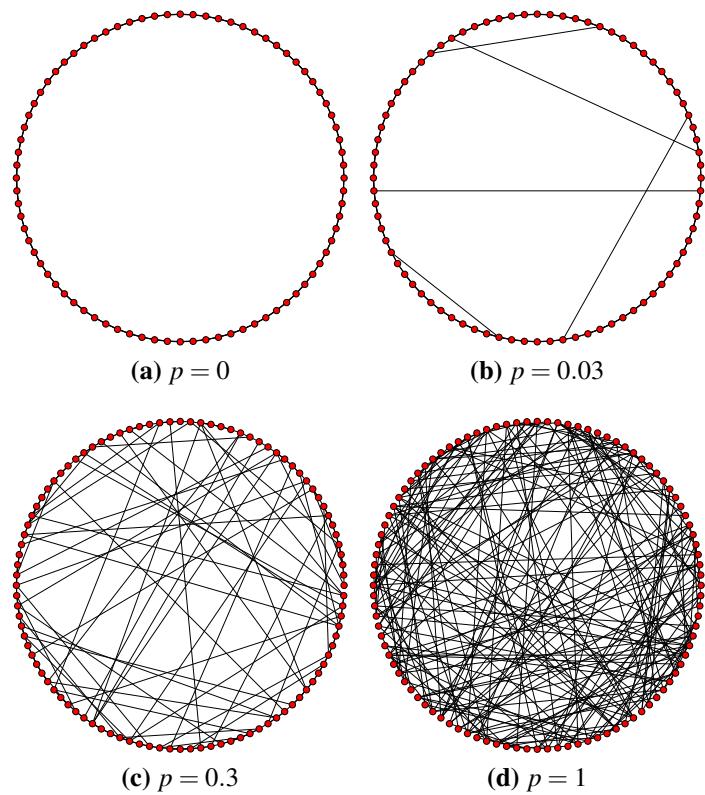


Figura 11: Generazione di grafi con il modello Watts-Strogatz a differenti p di rewiring.

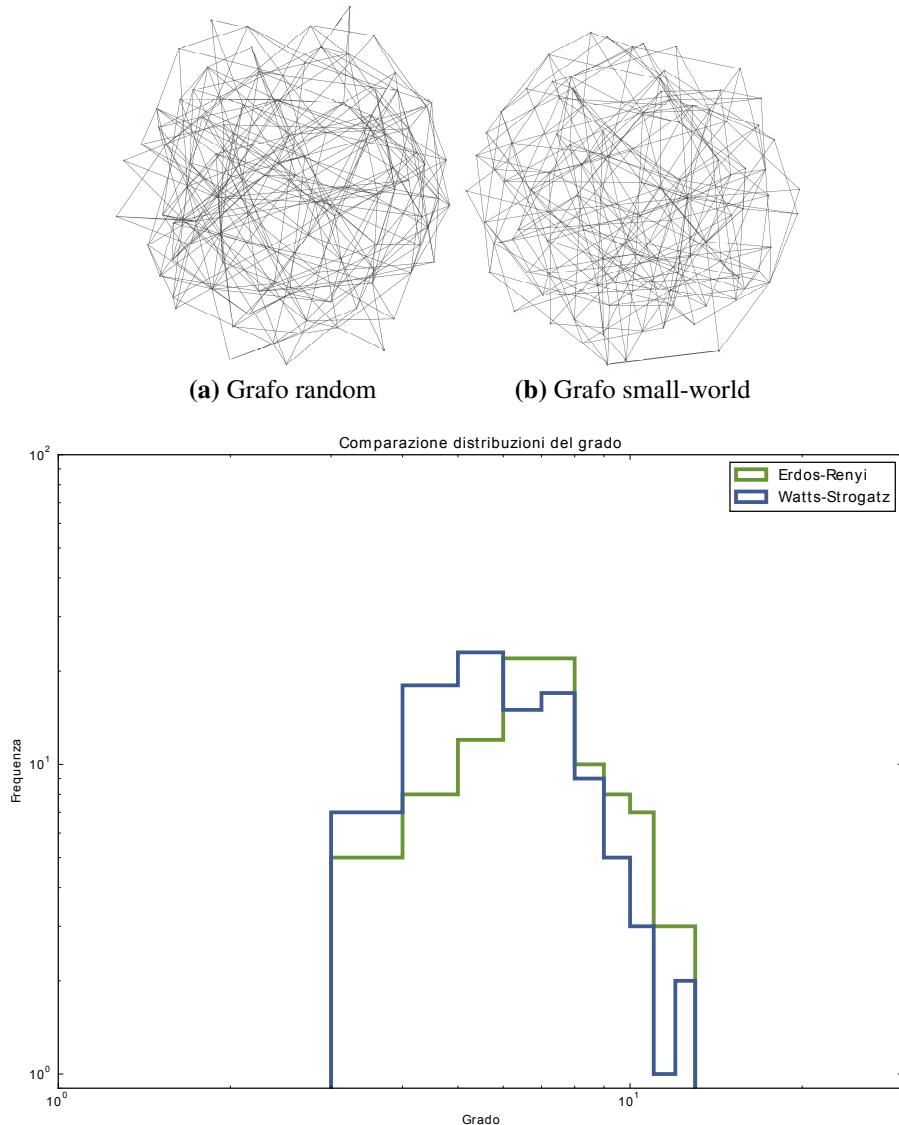


Figura 12: Confronto tra due grafi da 100 nodi: uno generato con il modello Erdos-Renyi ($p = 7\%$) e uno con il modello Watts-Strogatz partendo da un reticolo con nodi connessi ai primi 3 vicini e il 100% di probabilità di rewiring. A parte la somiglianza dei grafi, le distribuzioni del grado hanno simile comportamento.

Con questo metodo, a seguito del *rewiring* si ha il rischio che il grafo non sia più连通的. Ponendo $n > 1$ ciò può essere evitato, portando la probabilità di avere un grafo non连通的 quasi a zero già con $n = 2$. Con $p \rightarrow 1$ il grafo diventa simile a quello di una rete di Erdős-Renyi (v. Figura 12).

Per avere una rete tipica che abbia un numero di connessioni non troppo elevato, ma non così poco da rischiare di avere un grafo non连通的 a seguito dell'operazione di rewiring, possono essere considerati degli N e n tali che $N \gg n \gg \ln(N) \gg 1$. Con queste condizioni la distribuzione del grado $P(k)$ ha una forma gaussiana la cui media coincide con n , con σ più piccole per p basse, tendente a una delta di Dirac per $p \rightarrow 0$.

Infatti, mentre con $p = 0$ si ha un reticolo con grado uguale per ogni nodo, l'operazione random di rewiring introduce una casualità sulla $P(K)$ ben descritta da una gaussiana per N grandi, che però ha una larghezza sensibilmente inferiore a quella di

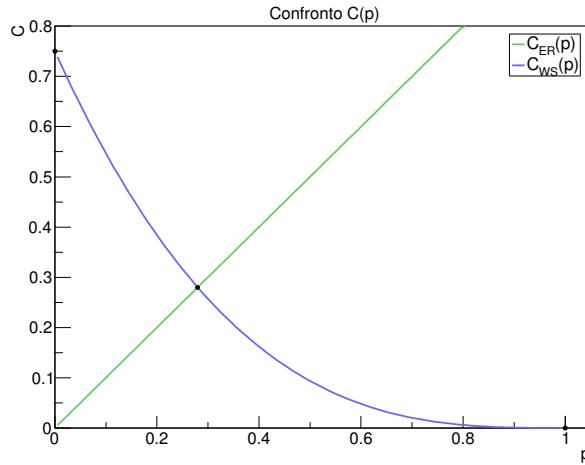


Figura 13: Confronto tra i coefficienti di clustering teorici dei due modelli, al variare del rispettivo parametro p .

una rete random, portando a una rete ancora più omogenea.

Valori tipici di $\langle l \rangle$ per le reti reali sono ben descritti dal modello di Erdős e Renyi secondo l’equazione 1 (Albert e Barabasi 2002), cioè ci si aspetta scali in modo logaritmico fissato $\langle k \rangle$. In una rete generata secondo il modello di Watts-Strogatz le lunghezze dei cammini, fissato $n \equiv \langle k \rangle$, scalano in media in modo diverso al variare di p . Per p molto basso ($p \ll 1/nN$) le lunghezze caratteristiche sono proporzionali alla dimensione del grafo, il quale è ancora molto simile a una reticolato; abbastanza presto, per $p >> 1/nN$,¹ vi è invece un largo intervallo di p che vede già verificarsi il fenomeno small-world, con le lunghezze dei cammini che scalano come $\ln(N)$ in accordo con le reti reali.

L’altra caratteristica fondamentale di uno small-world è che abbia un clustering abbastanza alto in relazione a una rete puramente random. Per $p = 0$ il reticolato ha un $C(0)$ costante al variare di N ; all’aumentare di p , presa una tripletta chiusa, la probabilità che tutti e tre i link rimangano inalterati è $(1 - p)^3$ e i suoi due primi vicini hanno probabilità p^3 di vedere almeno uno dei loro link riarrangiato. Dato che $C = N_\Delta/N_\wedge$ e che N_\wedge rimane costante con il rewiring, si può porre

$$C(p) \propto N_\Delta(p) \Rightarrow C(p) \sim C(0)(1 - p)^3,$$

con $C(0) \sim 0.75$ per N grande. Ricordando che per il modello di Erdős-Renyi $C_{ER} = p$, e che per modellizzare una rete reale p solitamente è abbastanza piccola dato che $\langle k \rangle = Np$, risulta che per valori di $p \lesssim 0.25$ si hanno velocemente C_{WS} sensibilmente maggiori di C_{ER} (v. Figura 13).

Concludendo, un grafo è considerato small-world se ha contemporaneamente piccole distanze medie e, a differenza dei grafi random, una clusterizzazione relativamente elevata. Pertanto il modello di Watts e Strogatz descrive in maniera soddisfacente reti che abbiano queste caratteristiche, purché non siano scale-free.

¹ $p >> 10^{-4}$ per una rete con 10^3 nodi e $P(k)$ centrata in 10

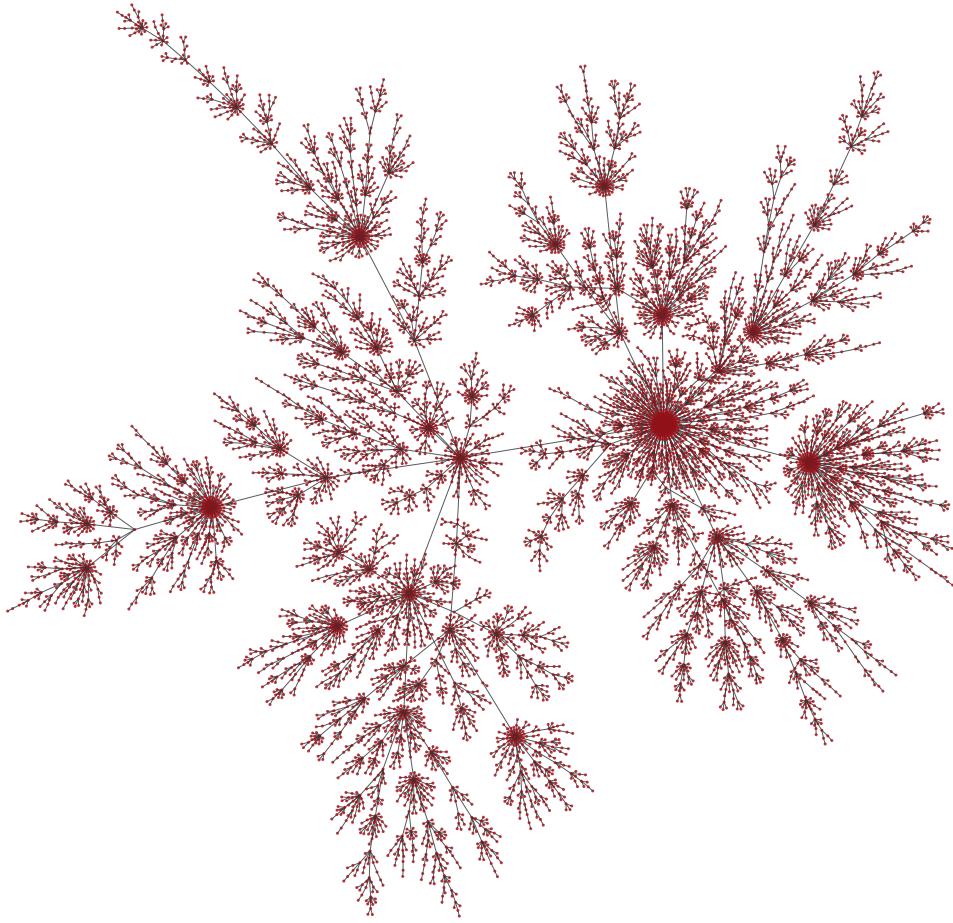


Figura 14: Esempio di grafo generato con il modello di Barabasi-Albert. Con $m = 1$ è evidente la struttura frattale, ma essendo un grafo ad albero il clustering è nullo.

3.4.3 Barabasi-Albert

Molte importanti reti reali di grandi dimensioni hanno la notevole caratteristica di avere un certo livello di invarianza di scala. Questa è manifestata da una distribuzione del grado che segue una funzione a legge di potenza $P(k) \sim k^{-\gamma}$, con γ compreso quasi sempre tra 2 e 3, in maniera sempre più esatta al crescere di k . Mentre le reti random possono riprodurre una $P(k)$ arbitrario, e quindi anche una *power-law*, ma non riescono a generare grafi connessi e con $\langle l \rangle$ non definibile (Albert e Barabasi 2002), le reti small-world riproducono bene le proprietà di clustering e cammini medi, ma non possono portare a $P(k)$ power-law. Serve pertanto un modello che riproduca piccoli diametri, grandi clustering e $P(k)$ a legge di potenza.

I modelli di Erdős-Renyi e Watts-Strogatz partono da una configurazione di nodi e poi distribuiscono o riarrangiano i link con una certa probabilità *uniforme* per tutti i link. Le reti reali, tuttavia, spesso partono da un certo numero piccolo di nodi e successivamente crescono. Il primo punto chiave del modello formulato da Barabasi e Albert è proprio il concetto di crescita: la rete parte a $t = 0$ con m_0 nodi e a ogni step temporale si aggiunge alla rete un nodo con un certo numero $m < m_0$ di link da assegnare agli altri

nodi esistenti. Il secondo riguarda il fatto che la probabilità di *attachment* di un nuovo nodo agli altri non è uniforme su tutti i nodi ma preferenziale: il *preferential attachment* dà quindi una maggiore probabilità $\Pi(k_i)$ di collegamento di un nodo nuovo a un certo nodo i in maniera proporzionale al suo grado, secondo la formula

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}.$$

La costruzione della rete avviene in modo dinamico, pertanto il grado di un nodo i sarà funzione crescente del suo tempo di vita nella rete, con una probabilità di attachment dei nuovi nodi a i anch'essa dipendente da t . Pertanto, con l'andare del tempo il grado di i aumenterà sempre più velocemente. Approssimando k_i a una variabile continua per t grandi, dato che

$$\frac{dk_i}{dt} = m\Pi(k_i) = m \frac{k_i}{\sum_{j=1}^{N-1} k_j} = m \frac{k_i}{2mt - m} = \frac{k_i}{2t - 1} \sim \frac{k_i}{2t},$$

dove m è il numero di link aggiunti per iterazione, ponendo quindi la condizione iniziale per il nodo i , $k_i(t_i) = m$

$$\Rightarrow k_i(t) = m \left(\frac{t}{t_i} \right)^{\frac{1}{2}}.$$

Ovviamente se il grado di un nodo è funzione di t , anche la $P(k)$ lo sarà. Definendo $P(k_i(t) < k)$ la probabilità che un nodo i abbia un grado minore di un certo valore k , la distribuzione del grado $P(k)$ può essere derivata ponendola uguale a $dP(k_i(t) < k)/dk$, ottenendo per $t \rightarrow \infty$:

$$P(k) \sim 2m^2 k^{-3}.$$

Il modello di Barabasi e Albert è un modello minimale, importantissimo perché il primo in grado di riprodurre un grafo che mostrasse un'invarianza di scala, ma che in alcuni aspetti mal si accorda con quelle reti reali con caratteristiche scale-free. Per cominciare la distribuzione del grado è una legge di potenza con esponente 3, ma le reti scale-free solitamente hanno un esponente inferiore, seppur maggiore di 2. Benché non sia possibile calcolare in modo analitico $\langle l \rangle$ e C per una rete di Barabasi-Albert, da simulazioni numeriche è possibile ottenerne quantità caratteristiche.

Il cammino medio risulta sottostimato rispetto alle reti reali e il coefficiente di clustering non è costante con l'aumentare di N come per le reti reali ma diminuisce, anche se più lentamente di una rete random² (Albert e Barabasi 2002). Per questo motivo il modello iniziale è stato integrato e reso più complesso.

²Ovviamente il coefficiente di clustering dipende dal numero m di link generati per ogni nuovo nodo. Con $m = 1$ il clustering è nullo dato che il grafo è un albero, mentre $C \neq 0$ per $m > 1$ e cresce all'aumentare di m , mantenendosi comunque al di sotto dei valori di reti reali con parametri simili.

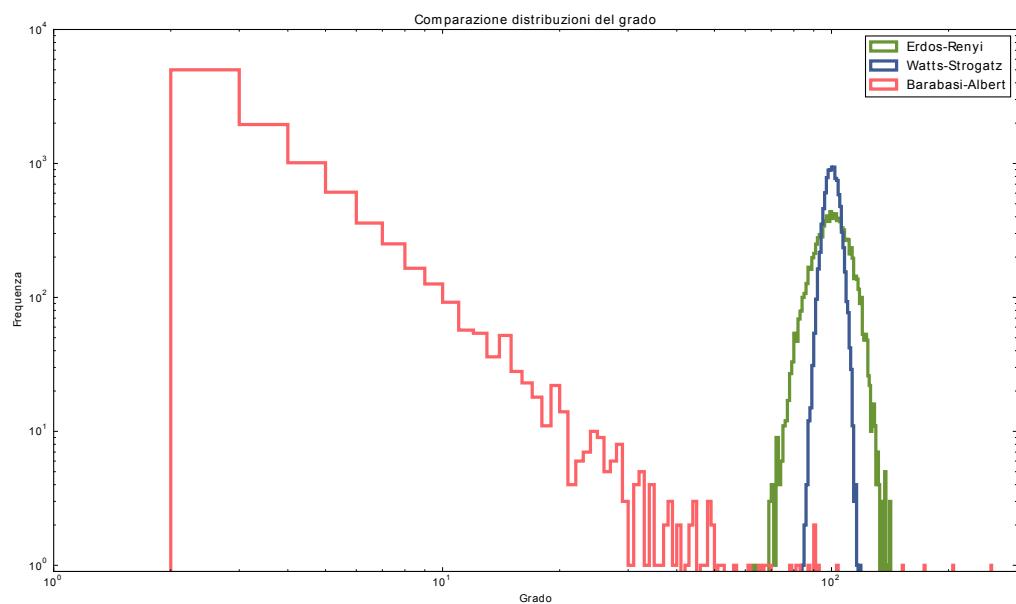


Figura 15: Confronto tra le distribuzioni del grado dei tre modelli esposti.

4 Teoria della percolazione

Per percolazione, in analogia con i liquidi, si intende il passaggio di informazione da un lato all’altro di una rete. Nello specifico, quando si fa uno studio percolativo di una rete, si cerca l’esistenza o meno di un cluster che ricopra l’intera rete (*spanning cluster*, o *giant cluster*). Il giant cluster è definito come quella componente di una rete le cui dimensioni sono dello stesso ordine della dimensione della rete complessiva (e quindi gli altri nodi o cluster minori hanno dimensioni di ordini inferiore), cioè il rapporto tra le sue dimensioni e le dimensioni dell’intera rete è ~ 1 .

Tipicamente uno studio percolativo su una rete random consiste nell’individuare, durante la costruzione della rete partendo da N nodi scollegati, la soglia percolativa p_c , cioè la probabilità di creazione link tale che compaia lo spanning cluster. Nel caso di una rete costruita con il modello di Barabasi-Albert la questione non si pone, dato che per definizione questa è composta da un solo cluster in crescita progressiva. Inoltre, dato che una rete di antenne cellulari nasce con una progettazione molto dettagliata in modo che venga costruita già oltre la soglia percolativa, nel caso in esame non è praticabile né interessante individuare p_c in costruzione. In questo caso ha più senso individuare la soglia percolativa per distruzione della rete.

Uno studio per distruzione può avere più approcci che portano a soglie diverse:

- se si parte da una rete con uno spanning cluster, rimuovendo link tra tutte le coppie di nodi connesse con una certa probabilità, si avrà una p_d oltre la quale la rete non percola più. In questo caso $p_{d_{link}} = 1 - p$ soltanto se si parte da un grafo completamente connesso.
- in alternativa si possono rimuovere i nodi in maniera random. In questo caso, rimuovendo un nodo si rimuovono in un colpo solo tutti i link a esso collegati, portanto quindi a una $p_{d_{nodo}} \neq p_{d_{link}}$. Si può dare una probabilità maggiore ai nodi con grado più alto per simulare un attacco mirato.
- in maniera opposta a un attachment progressivo, può essere usata un metodo di distruzione progressiva, nel quale si rimuove un nodo per volta e si simula la rimozione successiva con la rete risultante dalla rimozione precedente. Nel caso di una rimozione random, la differenza di questo metodo progressivo con quello diretto è poco significativa, ma risulta molto differente con l’attacco intenzionale, come spiegato in maniera più dettagliata nel paragrafo 5.2. Per rimozione progressiva dei nodi, che sia random o intenzionale, la soglia percolativa è espressa come la frazione critica f di nodi rimossi, rispetto al numero di nodi originale, oltre la quale non si ha più uno spanning cluster.

4.1 Soglia percolativa

Nel caso di rimozione di nodi da una rete, è interessante determinare quale è la frazione f di nodi che è necessario rimuovere per avere una frammentazione completa della rete. Un criterio per stabilire l’esistenza del giant cluster si basa sull’approssimazione che la rete sia abbastanza frammentata da poter trascurare i cicli. In questa situazione se esiste il cluster che copre tutta la rete è ridotto con buona approssimazione a un albero,

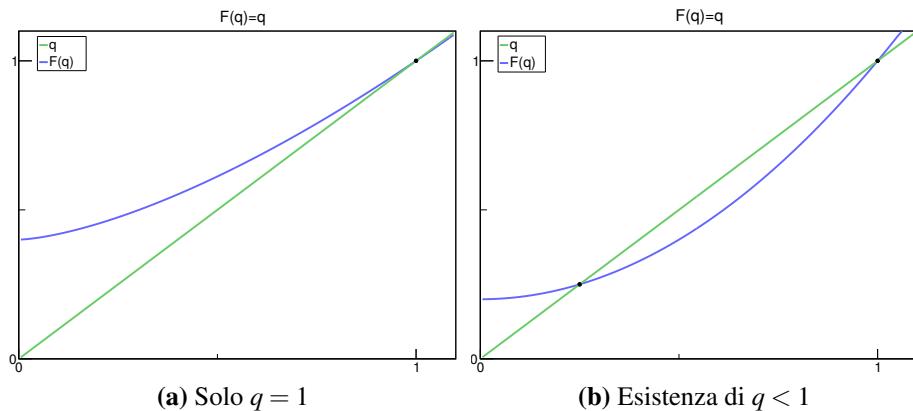


Figura 16: Confronto tra il caso in cui $F(q) = q$ ha solo la soluzione banale $q = 1$, e il caso in cui esiste una soluzione per $q < 1$.

e quindi si ha che, preso un nodo qualsiasi connesso al giant cluster, questo sia connesso a almeno un altro nodo (Cohen et al. 2000).

Definiamo quindi la probabilità q che un link scelto a caso non porti a un nodo che non faccia parte del giant cluster. Se $q < 1$ significa che esiste almeno un nodo che porti al giant cluster, e quindi il giant cluster esiste. L'intento è sapere quando è 1, per far ciò possiamo calcolare q definendola come la probabilità $P_{\rightarrow}(k)$ che un link scelto a caso porti a un nodo di grado k , per la probabilità q^{k-1} che gli altri $k - 1$ nodi non portino al giant cluster, sommato su tutti i possibili gradi:

$$q = \sum_k P_{\rightarrow}(k) q^{k-1}, \quad (2)$$

Se definiamo a sua volta $P_{\rightarrow}(k)$ come la probabilità di avere un nodo di grado k , per il numero di link possibili con cui possiamo ottenerlo (cioè il suo grado), allora $P_{\rightarrow}(k)$ diventa

$$P_{\rightarrow}(k) = \frac{P(k)k}{\sum_{k'} P(k')k'} = \frac{P(k)k}{\langle k \rangle}$$

$$\Rightarrow q = \frac{1}{\langle k \rangle} \sum_k P(k)kq^{k-1}$$

Questa è una funzione parametrica e ricorsiva. Chiamandola $F(q)$ e studiandola si ha che ha derivate prima e seconda positive, pertanto è sempre crescente e convessa; inoltre si ha che $F(1) = 1$ e $F(0) > 0$. Tutto ciò vale per qualsiasi parametro k .

Risolvendo l'equazione $q = F(q)$ si ottiene la condizione secondo cui il giant cluster non può esistere. Infatti, se esiste una soluzione oltre a quella banale per $q = 1$, significa che l'equazione 2 è verificata per valori di $q < 1$ e che quindi il giant cluster esiste. Come si può vedere dalla figura 16, si può adottare un metodo grafico: la retta e la curva si intersecano sempre in due punti, il primo con $F'(q) < q' = 1$ e il secondo con $F'(q) \geq 1$. Dato che uno dei due deve sempre essere $F(1) = 1$, se

$$F'(q=1) > 1$$

significa che esiste un altro punto di intersezione per $q < 1$ e quindi esiste il giant cluster. Svolgendo $d_q F|_{q=1} \geq 1$ si ottiene la condizione di esistenza dello spanning cluster in termini del grado, o criterio di Molloy e Reed (1995):

$$\frac{\langle k^2 \rangle}{\langle k \rangle} \geq 2. \quad (3)$$

Questo risultato è valido per ogni tipo di rete (Cohen et al. 2000). Un nodo con un grado iniziale k_0 avrà, dopo la rimozione di una frazione f di nodi, un grado k con una probabilità dettata da una distribuzione binomiale $B_{p,k_0}(k)$, e quindi la nuova distribuzione dei gradi sarà $P(k) = \sum_{k_0} P(k_0)B_{p,k_0}(k)$. Ricavando i nuovi $\langle k \rangle$ e $\langle k^2 \rangle$ si ottiene che la frazione di nodi rimossi necessaria perché si verifichi che $\langle k^2 \rangle / \langle k \rangle = 2$, e cioè il grafo sia completamente frammentato, è

$$f = 1 - \frac{1}{\frac{\langle k^2 \rangle}{\langle k \rangle} - 1} \quad (4)$$

Nel caso specifico di una rete costruita con il modello di Barabasi-Albert, questa avrà $P(k) \propto k^{-\alpha}$, che spazierà da un minimo m e un massimo K . In questo caso, ponendo $K \gg m \gg 1$, si può approssimare k a una grandezza continua e si ottengono due comportamenti a seconda del valore di α :

$$\frac{\langle k^2 \rangle}{\langle k \rangle} = c \frac{2 - \alpha}{3 - \alpha},$$

con

$\alpha > 3 \Rightarrow c \sim m$. Si ha una transizione allo stato completamente frammentato per valori di $f < 1$

$\alpha < 3 \Rightarrow c = c(K)$. La transizione dipende dal valore di K , per cui esiste per reti finite, ma per $K \rightarrow \infty f \rightarrow 1$.

Nel caso dell'attacco intenzionale la percentuale f è ottenuta per via numerica. Dato inoltre il peculiare metodo di attacco scelto, sarebbe necessario uno studio progressivo che a ogni iterazione tenga conto del taglio dei k più grandi e ricalcoli la $P(k)$ per ottenere il rapporto $\frac{\langle k^2 \rangle}{\langle k \rangle}$. Risulterebbe pertanto più conveniente effettuare simulazioni numeriche direttamente sui modelli (Cohen et al. 2001).

In ogni caso, a livello qualitativo, come nel caso di attacco random ci si aspettava una maggiore resistenza da parte delle reti scale-free, a causa del grande numero di nodi poco connessi che garantisce una $P(k)$ a legge di potenza con un opportuno α , nel caso dell'attacco intenzionale ci si aspetta una certa fragilità. Infatti le reti scale free sono costruite attorno ai nodi più connessi, si può supporre quindi che rimuoverli per primi provochi un deterioramento più veloce rispetto a una rete random con uguale grado medio (Albert e Barabasi 2002).

5 Network breakdown

Una volta osservate le distribuzioni del grado nelle reti delle quattro compagnie e nella rete complessiva formata da tutte le antenne comprese nell'area metropolitana di Roma, procediamo con lo studio percolativo. In riferimento al lavoro fatto da Albert, Jeong e Barabasi (2000), la scelta è stata di simulare due differenti scenari in cui i nodi della rete vengono disabilitati. Nel primo scenario si è ipotizzato un attacco intenzionale che cominciasse dai nodi con maggior grado, nel secondo una rimozione random.

Lo scopo è studiare l'andamento, in funzione della percentuale di nodi rimossi, del diametro D della rete e della dimensione del cluster più grande (ipotizzando, o meglio verificando, che esso sia il Giant Cluster della rete) rapportata al numero totale di nodi sopravvissuti. A seconda di come la rete si comporterà, sarà possibile dedurne la robustezza nei due scenari, in modo tale da poter confrontare meglio tale comportamento con quello di una rete scale-free o random, di pari grado medio $\langle k \rangle$.

In tutti e cinque i campioni di rete che abbiamo analizzato sono stati conteggiati gli andamenti di alcune grandezze topologiche e statistiche di interesse: dimensione del giant cluster, D e $\langle l \rangle$, coefficiente di clustering globale C , $\langle k \rangle$ e $\langle k^2 \rangle / \langle k \rangle$. I grafici ottenuti sono stati messi a confronto a quelli ottenuti con le reti generate secondo i modelli.

La rete reale ha ovviamente delle contromisure per evitare la caduta delle comunicazioni. Le antenne trasmettono segnali tra loro su due bande di frequenza: una *user-side*, dedicata alle normali trasmissioni tra utenti del servizio, e una dedicata a un complesso sistema di feedback gestito da degli *hub* (grosse antenne con raggio sui 20 km). Questa struttura gerarchica permette, nel caso di caduta di una antenna o di un improvviso eccessivo carico in una zona circoscritta, che gli hub gestiscano potenza e capacità delle antenne circostanti mentre vengono inviati tecnici per un intervento sul luogo.

Questo sistema ha un certo tempo di reazione. L'analisi da noi svolta pertanto suppone che la caduta della rete avvenga in un tempo inferiore, in una sorta di approssimazione adiabatica. Inoltre, la sola caduta degli hub sarebbe già sufficiente a compromettere seriamente l'integrità della rete (le antenne avrebbero difficoltà a coordinare le comunicazioni tra loro), ma nella nostra ipotesi di mesh-network distribuita ci interessano soltanto le comunicazioni nelle frequenze user-side. Usando questo modello semplificato siamo riusciti a ottenere alcune informazioni su una ipotetica rete wireless di questa natura.

5.1 Speedup del codice (parte 1: multiprocessing)

Per le simulazioni di attacco intenzionale e random failure sulle reti in esame e lo studio approfondito dell'andamento delle grandezze statistiche e topologiche durante l'analisi percolativa abbiamo inizialmente utilizzato *networkx*, una libreria di Python piuttosto completa dedicata alle reti.

Tuttavia questa si è rivelata una scelta sbagliata, in quanto *networkx*, pur essendo molto completa e semplice da usare, non è particolarmente ottimizzata dal punto di vista dell'efficienza di calcolo. Questo perché è interamente scritta in Python, anche per le sue routine più interne e non implementa alcun tipo di parallelismo *built-in*.

Creare, manipolare e disegnare i grafi sono operazioni semplici e immediate, ma quando si cercano di usare le funzioni più pesanti dal punto di vista del calcolo, come

per esempio quella per determinare il diametro della rete, networkx appare del tutto non soddisfacente.

Per esempio, il calcolo di diametro, average path lenght, coefficiente di clustering e dimensioni del giant cluster impiegava 20 secondi per una singola iterazione con la rete Tre (la più piccola, circa 1400 nodi), 50 secondi per Tim e Vodafone (circa 1800 nodi entrambe) mentre con la rete Wind (la più grande, con circa 2350 nodi) impiegava ben 2.5 minuti. Questo dato, moltiplicato per i 100 passi richiesti, porta ad una previsione di più di 6 ore di calcolo per l’analisi percolativa di tutte le singole reti.

Abbiamo osservato che i tempi scalano all’incirca quadraticamente col numero di nodi, per cui per lo studio della rete complessiva di tutta Roma sarebbero richiesti 20 minuti per una iterazione e quasi 30 ore di calcolo per l’analisi completa. Queste considerazioni rendevano di fatto impraticabile questa strada.



Dato che i tempi di calcolo richiesti per l’analisi percolativa risultavano spropositati, soprattutto nel caso di random failure, abbiamo cercato dei modi per velocizzare il calcolo ed aggirare il problema.

In primis si è cercato di evitare operazioni cicliche sui vettori dinamici, come `pop()` e `append()` e di utilizzare sempre librerie python ottimizzate e internamente scritte in C, come ad esempio `numpy`. Già questo permette operazioni su grossi vettori con un incremento di prestazioni rispetto al python puro di almeno il doppio.

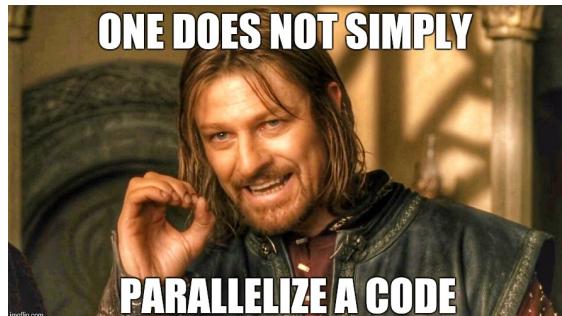
Inoltre si è tentato di sfruttare i diversi core della macchina, facendo degli esperimenti di rudimentale calcolo parallelo. In pratica si è cercato di rendere il codice non sequenziale, in modo da usare funzioni del tutto indipendenti e poter scrivere tutto in termini di funzioni `map()` per poi utilizzare la libreria built-in `multiprocessing` e fare una prima parallelizzazione.

```
import multiprocessing
cpus = multiprocessing.cpu_count()
pool = multiprocessing.Pool(processes=cpus)
pool.map(function, array)
```

Abbandonare il paradigma sequenziale rendendo le funzioni totalmente indipendenti tra loro fa sì che esse possano girare in contemporanea riducendo il tempo di calcolo, ma nel contempo aumenta il quantitativo di RAM richiesta dal programma, quasi dello stesso fattore. Pertanto bisogna fare attenzione a far girare il programma su una macchina adeguata sotto tutti i punti di vista.

Purtroppo però nel nostro caso questo approccio non si è rivelato adeguato. Nel fare i tentativi vedevamo che i processori lavoravano insieme al 100% per un primo periodo, per poi ibernarsi in una eterna fase di stallo al 20% del carico, probabilmente per

incongruenze nel messaging tra le varie istanze di python. Non possedendo sufficienti conoscenze sul calcolo parallelo e il cloud computing per poter risolvere velocemente il problema, abbiamo dunque abbandonato questo approccio.



5.2 Strategie di attacco

L'esigenza di provare a parallelizzare il codice ci ha portato a due versioni differenti della funzione di attacco intenzionale: una sequenziale e una parallela. Entrambe le varianti rimuovono circa l'1% dei nodi per volta:

- la funzione sequenziale elimina l'insieme dell'1% dei nodi più importanti, analizza la rete e calcola il prossimo insieme di 1% di nodi da rimuovere;
- la funzione parallela ha invece un comportamento più adiabatico: guarda il grafo iniziale e fa una classifica assoluta dei nodi per importanza, rimuovendone di volta in volta l'1% in maniera ordinata;

Possiamo supporre che un reale attacco alla rete sia portato avanti con modalità adiabatiche: un ipotetico terrorista piazza nella notte delle cariche esplosive sotto le antenne che ritiene fondamentali e poi le fa scoppiare durante il giorno tutte in una volta. A titolo di esempio, possiamo immaginare che l'attacco si consideri riuscito quando metà dell'area metropolitana rimane isolata senza possibilità di comunicare. Questo è il parametro finale in base al quale viene scelta la sequenza di antenne da far esplodere.

Ma chi garantisce che questa sequenza coincide con gli N nodi di grado maggior all'istante iniziale? In altre parole: è questa una strategia *ottimale*?

La risposta è no! La strategia ottimale è quella che, a parità di risultato, coinvolge il minor numero di antenne fatte esplodere, anche per minimizzare il rischio di essere scoperto nella notte mentre piazza l'esplosivo in giro per la città.

Tale strategia può essere evidenziata solo da una simulazione **sequenziale**, fatta facendo evolvere la rete passo per passo, secondo una strategia di *steepest descent*. Questo garantisce di trovare la strategia ottimale, soprattutto se vengono inclusi anche gli effetti a cascata dovuti all'overload e alla saturazione di banda. Dal grafico sottostante si evince comunque che non c'è molta differenza tra procedere con un campionamento all'1% o eseguire una simulazione di attacco nodo per nodo, per cui ad ogni modo il costo computazionale complessivo non risulta eccessivo.

Nel caso di random failure (escludendo gli effetti di saturazione a cascata) non c'è invece alcuna differenza tra l'approccio sequenziale e quello parallelo, rendendo il secondo algoritmo preferibile nel caso si voglia sfruttare tutta la potenza delle moderne cpu multicore.

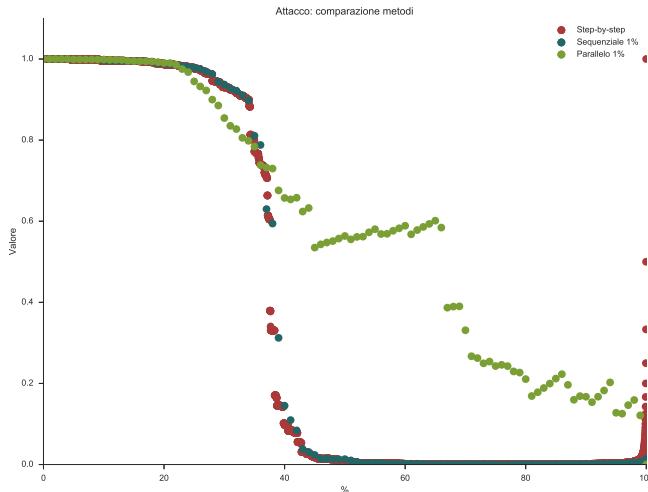


Figura 17: Confronto sugli effetti dei tre metodi di attacco sulla dimensione del giant cluster.

Riassumendo:

- Nel caso di intentional attack è doveroso usare un approccio sequenziale.
- Nel caso di random failure è consigliabile utilizzare un approccio parallelo.

5.3 Speedup del codice (parte 2: graph-tool)

Come già detto la prima soluzione al problema dei tempi è stata tentare una parallelizzazione del codice in Python: dato che networkx non supporta il calcolo parallelo interno, si è tentato di parallelizzare le funzioni di attacco intenzionale e random failure. Abbiamo però mostrato come alcuni degli algoritmi siano intrinsecamente sequenziali.

Un possibile metodo per ovviare a ciò sarebbe potuto essere generare e salvare in memoria tutte le reti a tutte le iterazioni e successivamente usare delle funzioni parallelizzabili per l'analisi. Tuttavia con networkx la richiesta di tempo di calcolo rimane comunque alta e si aggiungerebbe anche il problema della memoria RAM, dello spazio su disco e dei tempi di lettura e scrittura.

L'unica via praticabile è stata ripiegare su graph-tool: un'altra libreria meno user-friendly ma estremamente più efficiente e soprattutto impostata di default sul calcolo parallelo. Le funzioni di analisi topologica e statistica sono molto più veloci rispetto a networkx, poiché implementate in C col *template programming* ed ottimizzate in fase di compilazione. Inoltre la possibilità di usarle nativamente in parallelo su 4 core e 8 thread ci ha permesso di calcolare le singole funzioni con una velocità più di 60 volte superiore a prima.

Considerato ciò, il codice per lo studio percolativo delle reti è stato convertito (con non poco sforzo) per poter fare uso della nuova libreria. Alla fine, per l'analisi percolativa completa di tutte le reti, ovvero con lo studio in funzione dei nodi rimossi delle quantità:

- diametro D
- average path lenght $\langle l \rangle$

- clustering coefficient C
- average degree $\langle k \rangle$
- dimensioni relative del giant cluster
- criterio di soglia percolativa (andamento di $\langle k^2 \rangle / \langle k \rangle$)

da una previsione iniziale di più di 24 ore di tempo di calcolo si è passati a circa 3/4 d'ora, con uno speedup di circa un fattore 30.



5.4 Il punto di partenza

Le reti da cui è partito lo studio percolativo sono le 5 definite e costruite nel precedente paragrafo. Le grandezze iniziali erano:

| Rete | N | GC% | D | $\langle l \rangle$ | C | $\langle k \rangle$ | $C/\langle k \rangle$ | $\langle k^2 \rangle / \langle k \rangle$ | f% |
|----------|------|-----|---|---------------------|------|---------------------|-----------------------|---|------|
| Tim | 1756 | 1 | 2 | 1.96 | 0.89 | 64.3 | 0.014 | 359 | 99.7 |
| Vodafone | 1771 | 1 | 2 | 1.97 | 0.87 | 52.9 | 0.016 | 243 | 99.6 |
| Wind | 2365 | 1 | 2 | 1.96 | 0.89 | 97.3 | 0.009 | 478 | 99.8 |
| Tre | 1395 | 1 | 2 | 1.97 | 0.88 | 41.4 | 0.021 | 239 | 99.6 |
| Roma | 7287 | 1 | 2 | 1.97 | 0.89 | 239.1 | 0.004 | 1353 | 99.9 |

5.4.1 Random failure

La prima cosa da notare è che le reti delle singole compagnie hanno un rapporto coefficiente di clustering-grado medio in linea con molte altre reti reali (Albert e Barabasi 2002), mentre stranamente la rete costruita con tutte le antenne non vede aumentare particolarmente il suo clustering. Inoltre ci si aspetta che le reti siano tutte particolarmente resistenti sotto random failure, comportamento che sembra più simile a quello di una rete scale free. Tuttavia una rete random con un grado medio alto ha comunque un f molto alto. Per esempio, una rete random avente $N = N_{Tim}$ e $\langle k \rangle = \langle k \rangle_{Tim}$, avrebbe

una f critica di poco inferiore: $\sim 98\%$. Infatti, essendo per una rete random $\langle k \rangle = Np$, perché sia uguale a $\langle k \rangle_{Tim}$ con ~ 1800 nodi, deve essere $p \sim 3.7\%$,

$$\begin{aligned} \Rightarrow \sigma^2 &= Np(1-p) = \langle k^2 \rangle - \langle k \rangle^2 \sim 62 \\ \Rightarrow \frac{\langle k^2 \rangle}{\langle k \rangle} &= \frac{Np(1-p) - (Np)^2}{Np} \sim 1 + Np = 1 + \langle k \rangle \sim 65 \\ \Rightarrow f &\sim 1 - \frac{1}{65} \sim 98\% \end{aligned}$$

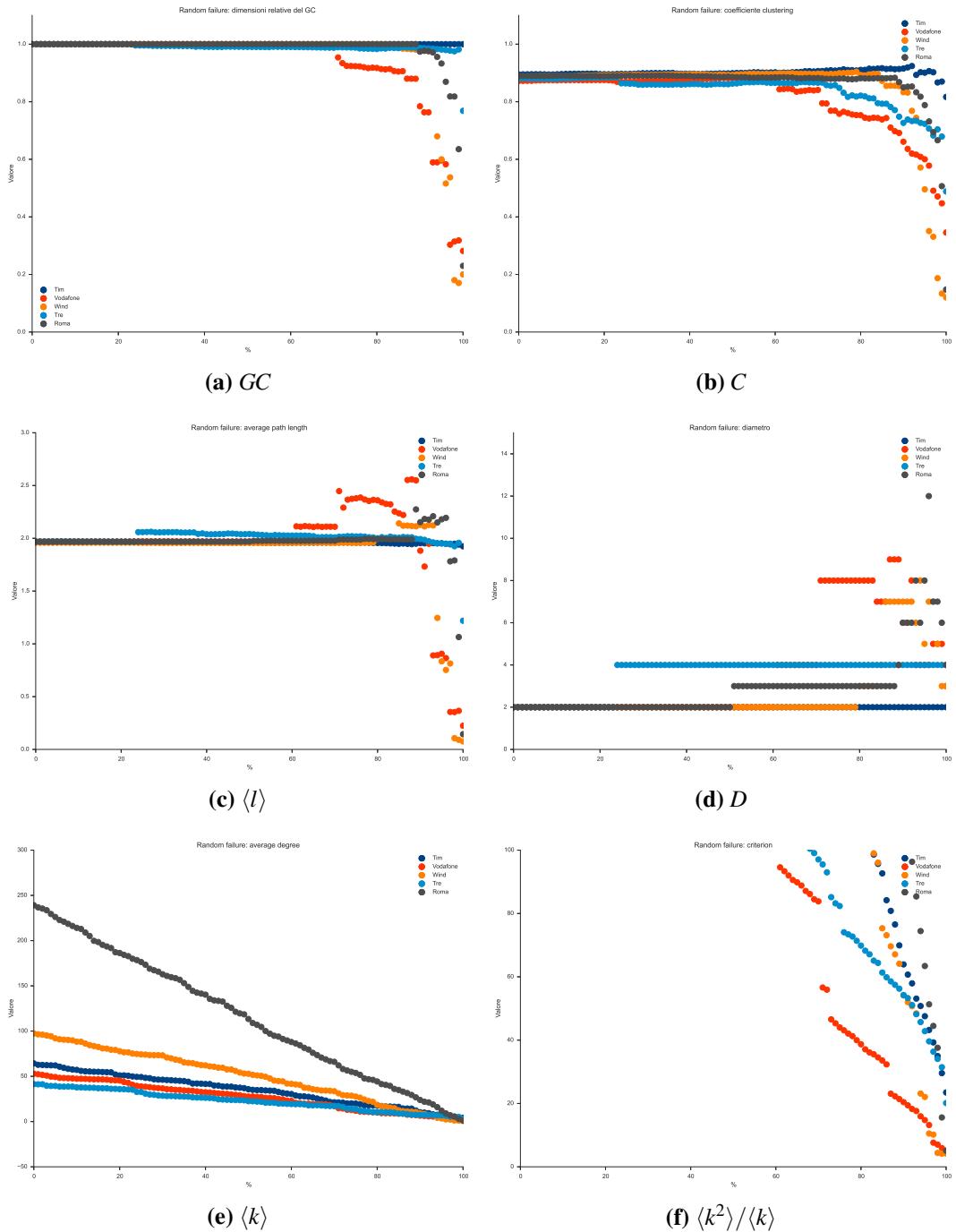
Con una rete random costruita con i parametri della rete totale si ottiene $f = 99.5$. Valori quindi che si distinguono poco dalla soglia per una rete power-law finita, anche nel caso in cui l'esponente sia di poco maggiore di 3: $f \sim 1 - \frac{1}{\frac{\alpha-2}{\alpha-3}k_{max}} > 90\%$ già per un grado massimo di ~ 10 .

5.4.2 Attacco intenzionale

Nello scenario di attacco intenzionale ci si aspetta una veloce frammentazione della rete. I cluster diverranno rapidamente più piccoli fino a frammentarsi del tutto entro pochi punti percentuali di nodi rimossi. Nel caso di una rete fortemente connessa come quella in esame ci si aspetta una resistenza maggiore, ma comunque una soglia percolativa bassa (approssimativamente entro il 50%). Se la rete è di tipo scale-free dovrebbe essere più fragile ad attacchi di questo tipo: mentre in una rete random i nodi più connessi sono solo una coda della distribuzione del grado, una rete a power-law ha in proporzione molti più nodi molto connessi.

Dal punto di vista del diametro della rete, levando i nodi più connessi ci si aspetta che esso aumenti, fino a quando la rete diventa tanto frammentata da essere costituita da clusters con pochissimi nodi. Oltrepassata la soglia di frammentazione quindi il diametro dei clusters più grandi decrescerà rapidamente a zero.

5.5 Risultati

**Figura 18:** Risultati per rimozione random

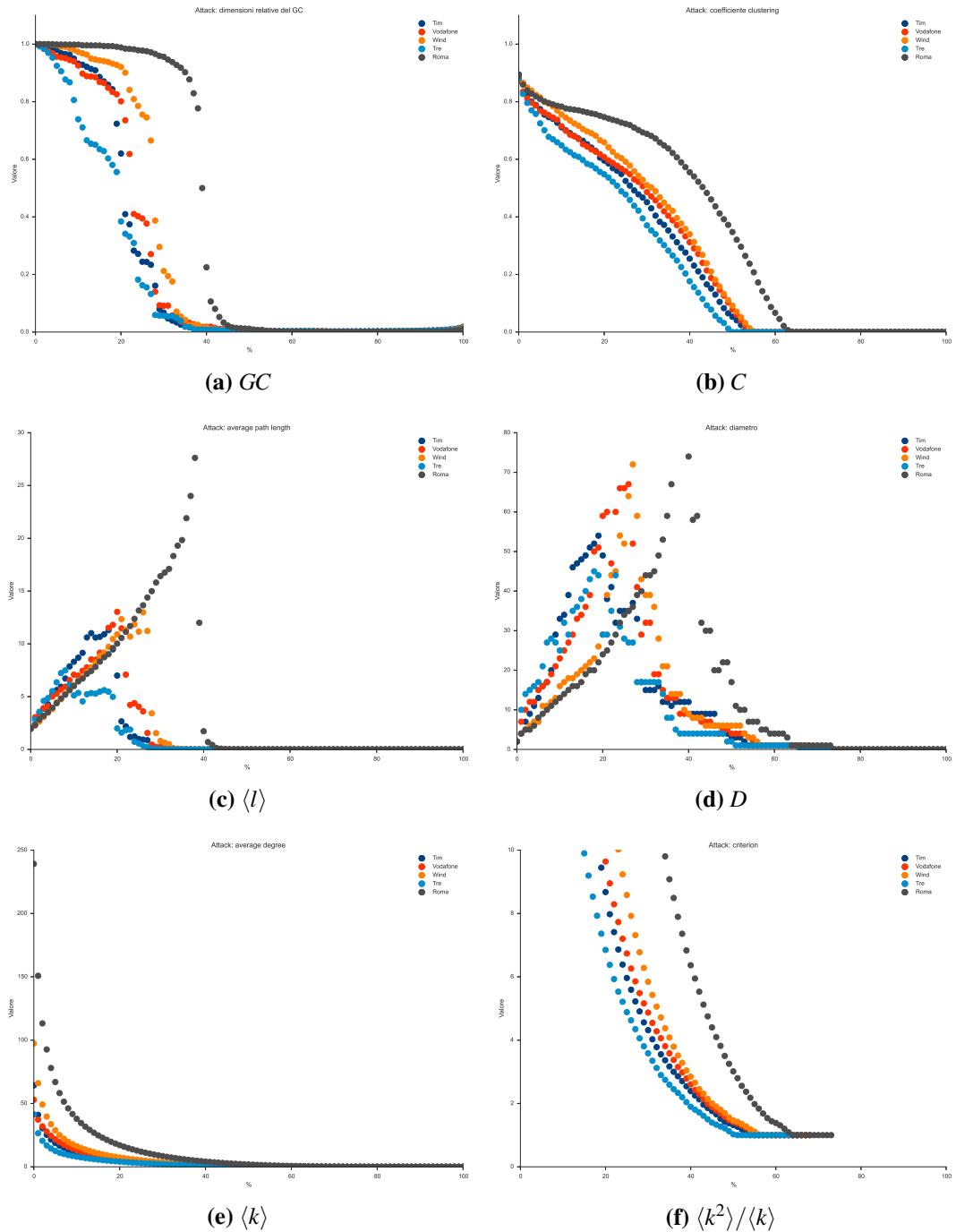


Figura 19: Risultati per rimozione con massima efficienza

6 Conclusioni

Come abbiamo visto, in regime di alta connettività, dal punto di vista percolativo si notano poco le differenze tra reti generate secondo un modello scale-free o random. La differenza si nota quando il grado medio della rete è più basso. Per esempio una rete random con $\langle k \rangle \sim 2$ ha $f \sim 50\%$, mentre una rete scale free con stesso $\langle k \rangle$ ha $f \sim 80\%$. Una spiegazione qualitativa sta nel fatto che con una distribuzione del grado a legge di potenza, i nodi meno connessi sono in larga maggioranza, quindi è meno probabile che un attacco random riesca a destrutturare la rete, rispetto a un grafo che abbia una $P(k)$ poissoniana.

Un'altra cosa da notare è che a parità di N , una rete di Barabasi-Albert è molto più connessa di una di Erdos-Renyi. Nell'esempio sopra, infatti, con $N = 100$ una la prima ha ~ 400 link, mentre la seconda il doppio; per avere lo stesso numero di link la rete random ha bisogno di una $p \Rightarrow \langle k \rangle$ doppi. Questa differenza va attenuandosi con N più grandi.

Uno studio percolativo su reti reali, quindi, non aiuta a identificare una eventuale invarianza di scala per reti molto grandi e connesse. Tuttavia può essere decisiva per reti più piccole, come piccole comunità, reti locali, proteine e alcuni ecosistemi: nonostante la bassa statistica, il comportamento di reti con pochi nodi possono essere distinti in maniera significativa da uno studio percolativo.

Nel caso di reti il cui funzionamento dipende dalla capacità dei nodi di gestire un certo carico, uno studio percolativo dovrebbe anche considerare l'eventualità che con la rimozione di alcuni nodi, altri vadano in sovraccarico e si scolleggino dalla rete (Motter 2002, Zhao 2004 e 2005, Wang 2009). È il caso delle reti comunicative, come quelle telefoniche cellulari analizzate, o una ipotetica rete mesh costruita con i router WiFi, ma anche ovviamente reti di distribuzioni elettriche, o alcuni ecosistemi dove una catena alimentare che veda la scomparsa di una specie causa una eccessiva riproduzione di una specie predata, che a sua volta porta altre specie all'estinzione.

Una possibile interessante applicazione dei metodi e degli strumenti usati per analizzare la rete di antenne di Roma potrebbe essere lo studio di città molto più grandi e densamente popolate (come New York o Tokyo), ma servirebbero capacità di calcolo e di memoria molto più grandi di quelle a nostra disposizione. Un altro possibile studio potrebbe essere verificare la possibilità di creare una rete distribuita e aperta con i router WiFi. Avendo questi un raggio molto inferiore delle antenne cellulari, la rete sarebbe molto simile a un reticolato, quindi benché uno studio percolativo possa essere interessante, non lo sarebbe dal punto di vista delle reti complesse. Una rete di questo tipo sarebbe comunque un nuovo concetto di Internet, che potrebbe avere interessanti sviluppi su reti di altro tipo, per esempio sociali, correlate a essa.

Bibliografia

- Albert, R. e A. L. Barabasi (2002). "Statistical mechanics of complex networks". In: *Reviews of modern physics* 74, p. 47.
- Albert, R., H. Jeong e A. L. Barabasi (2000). "Error and attack tolerance of complex networks". In: *Nature* 406, p. 378.
- Barabasi, A. L. e R. Albert (1999). "Emergence of scaling in random networks". In: *Science* 286, p. 509.
- Cohen, R. et al. (2000). "Resilience of the Internet to Random Breakdowns". In: *Physical Review letters* 85, p. 4626.
- (2001). "Breakdown of the Internet under Intentional Attack". In: *Physical Review letters* 86, p. 3682.
- Erdős, P. e A. Rényi (1959). "On random graphs I". In: *Publicationes Mathematicae Debrecen* 6, p. 290.
- Molloy, M. e B. Reed (1995). "A critical point for random graphs with a given degree sequence". In: *Random structures and algorithms* 6, p. 161.
- Motter, A. E. e Y. C. Lai (2002). "Cascade-based attacks on complex networks". In: *Physical Review E* 66, p. 065102.
- Mozilla Location Service*. Mozilla Foundation. URL: <https://location.services.mozilla.com/>.
- OpenCellID*. ENAiKOON. URL: <http://opencellid.org/>.
- Wang, J. W. e L. L. Rong (2009). "Cascade-based attack vulnerability on the US power grid". In: *Safety Science* 47, p. 1332.
- Watts, J. e H. Strogatz (1998). "Collective dynamics of 'small-world' networks". In: *Nature* 393, p. 440.
- Zhao, L., K. Park e Y. C. Lai (2004). "Attack vulnerability of scale-free networks due to cascading breakdown". In: *Physical Review E* 70, p. 035101.
- Zhao, L., K. Park, Y. C. Lai e N. Ye (2005). "Tolerance of scale-free networks against attack-induced cascades". In: *Physical Review E* 72, p. 025104.