Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea Triennale in Informatica

LA VISUALIZZAZIONE NELL'ANALISI DEI SOCIAL NETWORK

Tesi di Laurea in Basi di Dati

Relatore:
Chiar.mo Prof.
Danilo Montesi

Presentata da:
Francesca Guadagnini

Sessione I
Anno Accademico 2011/2012

It's more fun to be a pirate than to join the navy.
S. Jobs

Indice

Capitolo 1

Introduzione

Lo studio dei Social Network si è rivelato negli ultimi anni di grande utilità: monitorare campioni di persone e le loro relazioni ha rivelato comportamenti e abitudini mai notate prima. Internet infatti, è fatta prima di tutto di persone, e le persone creano tutto il contenuto. I Social Network rappresentano un enorme database, e di conseguenza un enorme potenziale per ricerche di ogni tipo, fra cui ricerche di mercato, studi sociologici, reazioni della massa a breve termine e via dicendo. In alcuni casi sono nate addirittura rivoluzioni, dove un tweet di un utente ha scatenato una reazione a catena, e il movimento di milioni di persone ha portato un cambiamento concreto. Un Social Network mette persone in relazione fra di loro, e crea ragnatele di informazioni a prima vista invisibili, ma se correttamente esportate ed elaborate possono prendere forma in qualcosa di concreto.

Si possono fare studi sull'umore, osservando il comportamento di twitter e delle hash-tag più usate, e arrivare ad estrapolare il sentimento generale della popolazione usando software di analisi semantica, tenendo traccia dei cambiamento e dell'evoluzione delle attitudini emotive degli utenti. Si possono anche fare studi su malattie, epidemie, su chi vincerà le elezioni: qualunque dato, se in grande quantità, diventa fonte di informazioni. Una volta estrapolati i dati però il passo successivo è dargli una forma e renderli chiari, in poche parole: visualizzarli.

I grafi sono utilizzati nell'ambito informatico e matematico per il loro carattere versatile e descrittivo, e sono alla base delle analisi delle reti sociali. Un grafo è modificabile e maneggevole ed esistono software in grado di trasformare una massa di dati in uno schema dinamico, permettendo di studiare la possibile evoluzione di rami e nodi e di visualizzare in tempo reale come le relazioni si modificano nel tempo.

Il mio studio si concentra prevalentemente su Twitter e sulla visualizzazione del traffico che è stato prodotto durante il terremoto in Emilia Romagna il 20 maggio e i giorni successivi. Twitter si è rivelato in molti casi la fonte di notizie più veloce e sincera rispetto a qualunque quotidiano, e i contenuti creati direttamente dagli utenti rappresentano una vera e propria impronta del pensiero delle persone. Tramite Gephi, un software open-source, estrarrò i dati da Twitter esaminando il comportamento degli

utenti durante e dopo le scosse, modellando grafi e mostrando il potenziale che queste strutture hanno, assieme, ovviamente, ai Social Network e allo sharing di informazioni.

Capitolo 2

I Grafi

2.1 Definizione di Grafo

Quando parliamo di grafo, ci riferiamo ad una struttura che permette di schematizzare una grande varietà di situazioni. I grafi sono utilizzati per rappresentare insiemi di oggetti e le relazioni che intercorrono fra essi. Molti problemi di calcolo sono riconducibili a problemi su grafi, e possono essere risolti con tecniche sperimentate.
Formalmente, si dice grafo una coppia ordinata di insiemi $G = (V, E)$, con V insieme dei nodi ed E insieme degli archi, tali che gli elementi di E siano coppie di elementi di V. Due vertici *"u, v"* connessi da un arco *"e"* prendono nome di *estremi dell'arco*; l'arco *"e"* viene anche identificato con la coppia formata dai suoi estremi *(u, v)* .
I grafi sono formati da *nodi* e *collegamenti*, e si dividono in *orientati* e *non orientati*. [1]

2.1.1 Grafi non orientati

Nei grafi non orientati i collegamenti sono chiamati *spigoli (edges)*. I nodi vengono rappresentati da pallini e gli spigoli da tratti. Gli spigoli servono per rappresentare relazioni di simmetria fra i nodi.
In un grafo non orientato la connessione da u a v ha lo stesso significato di quella da v a u ($u \rightarrow v = v \leftarrow u$).

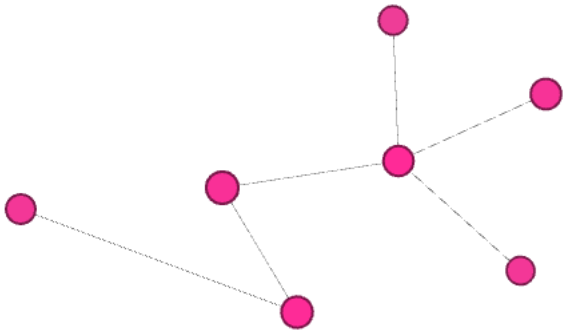


Figura 2.1: Un grafo non orientato

2.1.2 Grafi orientati

Per i grafi orientati si useranno invece frecce per collegare fra loro i nodi, in modo da evidenziare il senso di marcia del collegamento. Gli archi infatti sono caratterizzati da una direzione.
In particolare, l'arco è composto da una "testa" (rappresentata solitamente dalla punta di una freccia) che raggiunge un vertice in entrata, e una "coda", che lo lascia in uscita.

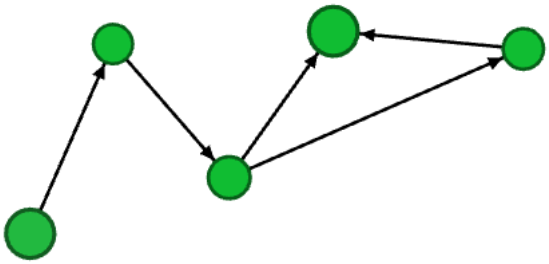


Figura 2.2: Un grafo orientato

2.1.3 Le connessioni

In un grafo, due vertici si dicono "connessi" se esiste un percorso che li collega. Al contrario, se non esiste nessun percorso si dicono "sconnessi". Un percorso, o cammino, è dato da una sequenza di nodi $(v_0, v_1, \dots v_n)$, collegati fra loro da una sequenza di archi $((v_0, v_1), (v_1, v_2), ..., (v_{n-1}, v_n))$.
I vertici v_0 e v_n si dicono estremi del percorso; se v_0 e v_n coincidono il cammino prende il nome di ciclo, o cammino chiuso. [1]

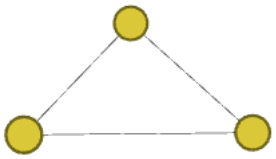


Figura 2.3: Un ciclo

Per introdurre il concetto di *componenti connesse* di un grafo, dobbiamo rifarci al concetto matematico di *relazione di equivalenza*, ovvero una definizione che si avvicina molto al concetto di similitudine. Una relazione di equivalenza è una relazione binaria tra elementi di un insieme A riflessiva, simmetrica e transitiva. La relazione " $\sim$ " si legge "è quivalente a". Valgono quindi le seguenti proprietà:

- $a \sim a$, $\forall a \in A$
- $a \sim b$ implica $b \sim a$, $\forall a, b \in A$
- $a \sim b$ e $b \sim c$ implica $a \sim c$, $\forall a, b, c \in A$

All'interno di un grafo le *componenti connesse* sono le classi di equivalenza dei vertici: a prima vista si possono identificare come quei gruppi di nodi connessi fra loro (sottografo), ma sconnessi da altri sottografi.

G' è un sottografo di G ($G' \subseteq G$) se e solo se $V' \subseteq V$ e $E' \subseteq E$.
Le classi di equivalenza formano una partizione, sono cioè insiemi disgiunti di A la cui unione coincide con tutto il grafo. [2]

Un grafo orientato si dice fortemente connesso se per ogni coppia di nodi u, v esiste almeno un cammino da u a v e da v a u.

2.2 Esplorazione di un grafo

Una visita di un grafo è una procedura sistematica per esplorare un grafo, visitando almeno una volta ogni suo nodo ed arco. Si può pensare ad esempio ai motori di ricerca,

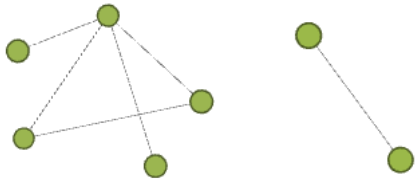
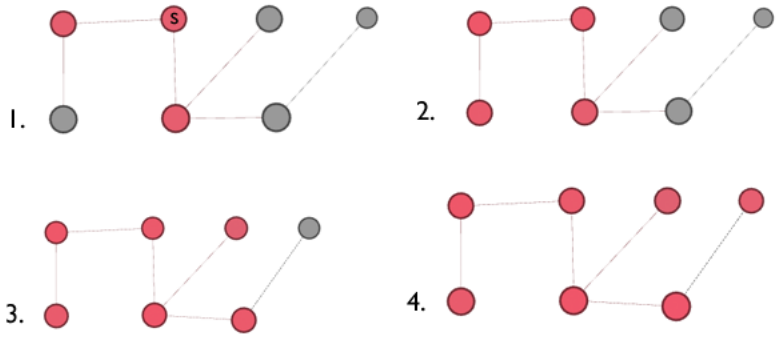


Figura 2.4: Un grafo con due componenti connesse

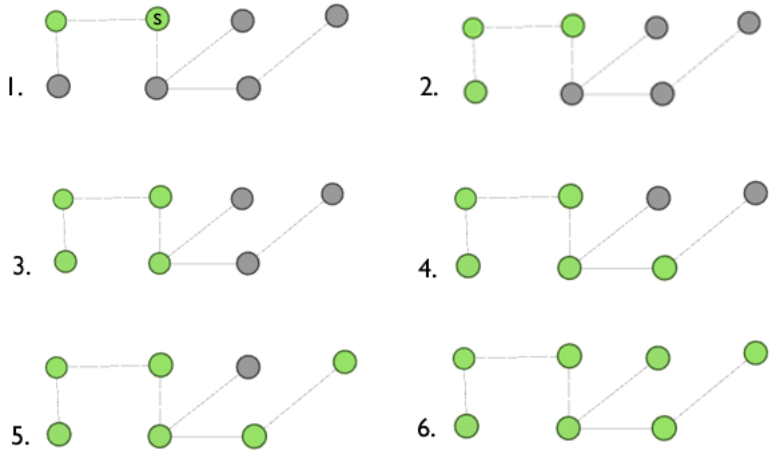
i quali grazie ai "web spider" raccolgono i dati necessari per rispondere alle richieste di un utente. Un web spider visita il grafo del Web, in cui le pagine web sono nodi e i link in esse contenuti sono archi. [3]

Due algoritmi celebri per la visita dei grafi sono *dfs (depth-first search)* e *bfs (breadth-first search)*: visita in profondità e visita in ampiezza. Nel bfs, dato un vertice sorgente s si esplorano sistematicamente tutti i vertici raggiungibili da s in modo tale da visitare tutti i vertici che hanno distanza k prima di iniziare a scoprire quelli che hanno distanza k+1. Il dfs invece consiste nella esplorazione sistematica di tutti i vertici andando in ogni istante il più possibile in profondità. [4]

Di seguito un esempio di grafo visitato con bfs:



E un esempio di visita dfs:



2.3 Base di dati a Grafo

Una Base di dati a grafo (o Database a grafo) usa nodi e archi per rappresentare e archiviare l'informazione.
E' un modello alternativo al modello relazionale che usa tabelle, ai database orientati al documento o ad altri archivi basati su colonne.
I Database a grafo sono spesso più veloci di quelli relazionali nell'associazione di set di dati, e mappano più direttamente le strutture di applicazioni orientate agli oggetti. Su grandi quantità di dati scalano più velocemente e non richiedono le tipiche e costose operazioni di unione (join). Dipendono meno dagli schemi ER, sono molto più adeguati per gestire dati mutevoli con schemi evolutivi. Al contrario, i database relazionali sono tipicamente più veloci nell'eseguire le stesse operazioni, ma su una grande quantità di dati. [1]

Capitolo 3

Gephi

3.1 Introduzione

Gephi è un visualizzatore interattivo e una piattaforma di esplorazione per tutti i tipi di reti e sistemi complessi, dinamici e grafi gerarchici.
E' disponibile per Linux, Mac e Windows, è gratuito e open-source.
E' uno strumento completo per la comprensione e l'elaborazione dei grafi, dove un utente può interagire con la sua rappresentazione grafica, manipolare le strutture, le forme e i colori per ordinarlo e renderlo più chiaro a colpo d'occhio. Uno degli obiettivi alla base di Gephi è quello di fornire una piattaforma che possa lavorare come un classico editor di immagini, ma sui grafi.
Offre numerosi algoritmi di manipolazione, per rendere possibile la scoperta di nuovi modelli e pattern. Possiamo pensarlo come uno strumento complementare alle statistiche tradizionali, infatti è in grado di semplificare, schematizzare e tradurre graficamente un insieme di dati attraverso le immagini. [5]

3.2 Funzionalità

3.2.1 L'interfaccia

L'interfaccia di Gephi si presenta come un classico editor di immagini, dove al centro troveremo l'oggetto da modificare e tutto intorno gli strumenti utilizzabili. Ci sono tre tab principali, ovvero tre schermate sulle quali lavorare. La prima, "Overview" è quella dove potremo visualizzare e manipolare il grafo. In "Data Laboratory" il grafo si presenta sotto forma di una tabella. Potremo quindi lavorare su colonne, su attributi, unire i campi che preferiamo e creare i legami (*arcs* o *edges*) fra i nostri nodi. Possiamo anche creare direttamente nodi: appariranno direttamente nella tab Overview. L'ultima schermata è "Preview", dove il grafo verrà reso piacevole graficamente e si potranno

apportare gli ultimi ritocchi ed esportarlo in pdf o in altri formati grafici vettoriali.

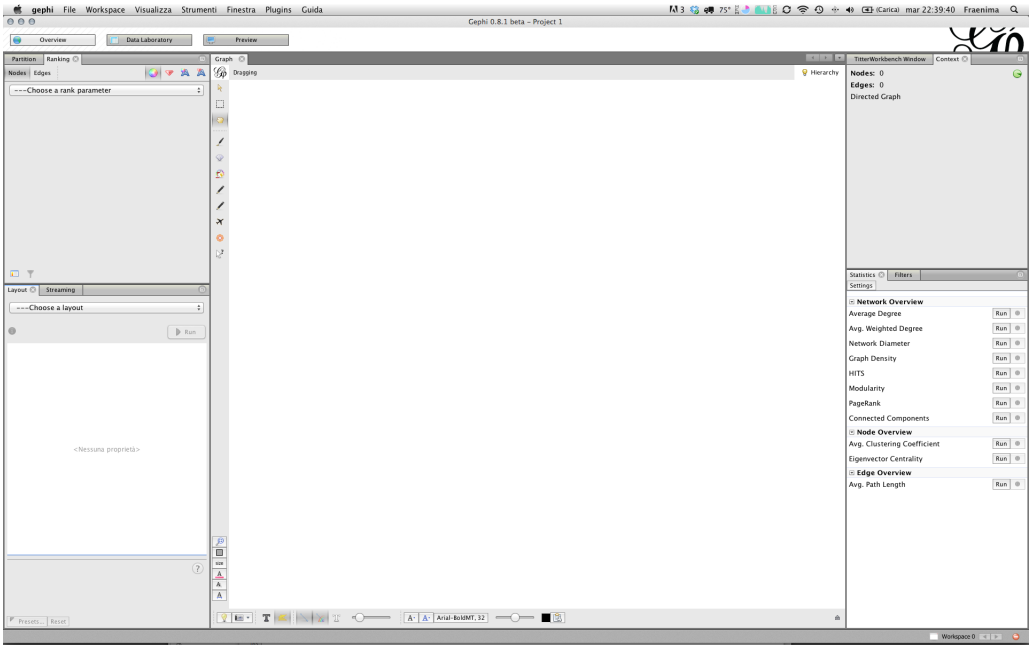


Figura 3.1: L'interfaccia di Gephii

I principali file importabili sono *.csv, .graphml, .gdf, .tlp, .vna*, archivi come *.zip* e *.rar* e ovviamente files*.gephi*.
Un file .csv importato si presenterà come un classico database, visualizzabile sia nella tab Data Laboratory sia sotto forma di grafo direttamente nella tab Overview. Ci sarà bisogno di un po' di lavoro prima di vedere un file .csv sotto forma di grafo, se si possiede infatti solo la tabella dei nodi, i collegamenti dovranno essere fatti tutti a mano.
Se invece si possiede già un file che rappresenta un grafo (.graphml o .gdf) possiamo aprirlo dalla schermata Overview e iniziare a lavorarci immediatamente. Appena importato, un grafo ovviamente non avrà un aspetto ordinato. Verrà visualizzato più o meno come un ammasso di nodi, i quali dovranno poi essere lavorati con i vari layout.

3.2.2 I layout

I layout sono gli algoritmi con i quali possiamo lavorare il grafo.

Force Atlas : layout Home-brew di Gephi (ovvero sviluppato in casa), è adatto a visualizzare le reti a invarianza di scala (per esempio reti sociali, internet, il numero di Bacon) e quei grafi dove i nodi sono per la maggior parte connessi fra di loro, o dove al più un nodo può essere raggiunto con pochi passi (Small-world Network). Si concentra

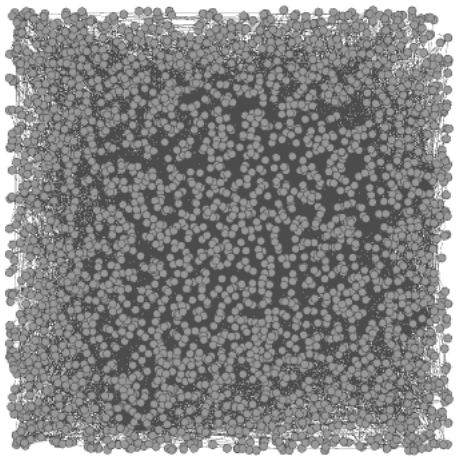


Figura 3.2: Un .gdf importato su gephi

sul rendere il grafo più leggibile possibile e offrire una interpretazione reale delle connessioni fra i nodi. E' abbastanza lento, ha complessità $O(n^2)$, dove n è il numero dei nodi, e riesce a lavorare su grafi da 1 a 10.000 nodi circa. Si possono settare parametri come "Repulsion strengh", ovvero quanto i nodi si respigeranno gli uni dagli altri o "Attraction strenght" dove al contrario si nodi si attireranno. "Autostab strengh" regolerà la lentezza con cui i nodi si sposteranno una volta lanciato l'algoritmo, "Gravity" attirerà tutti i nodi al centro per evitare una dispersione esagerata dei nodi sconnessi.

Fruchterman-Reingold : trasforma il grafo in un sistema di particelle di massa. I nodi verranno interpretati come particelle e gli archi come le spinte che si daranno l'un l'altra. L'algoritmo tenta di ridurre al minimo l'energia utilizzata del sistema fisico. E' molto lento, ha complessità $O(n^2)$ e lavora su grafi da 1 a 1000 nodi circa.

Yifan Hu Multilevel : si tratta di un algoritmo molto veloce, efficiente su grafi di grandi dimensioni. Organizza il grafo in cluster, ovvero gruppi di nodi connessi e simili fra di loro. La forza di repulsione da un nodo a un cluster di nodi è approssimata grazie all'algoritmo di Barnes-Hut. Si ferma automaticamente e ha complessità $O(n * log(n))$. Lavora su grafi da circa 100 a 100.000 nodi.

OpenOrd : Riesce a evidenziare meglio i cluster rispetto ad altri algoritmi separandoli e allontanandoli ancora di più; per questo motivo è ottimo su grafi di grandi dimensioni, fino a un milione di nodi. Su grafi più piccoli è consigliabile utilizzare un altro layout, perché troppo dispersivo. E' basato originariamente su Frutcherman-Reingold e

procede lavorando su iterazioni fisse (un parametro impostabile). Può essere lanciato in parallelo per velocizzare la computazione. E' molto veloce e ha complessità $O(n*log(n))$.

Force Atlas 2 : la versione migliorata di Force Atlas, complessità $O(n * log(n))$. Lavora più velocemente rispetto al suo predecessore, anche su grafi di grandi dimensioni, approssimando la repulsione fra i nodi grazie a Barnes-Hut, che a sua volta riduce la complessità dell'algoritmo.

Circular : Sistema i nodi su una circonferenza, ordinandoli dal più influente secondo un parametro a scelta, come per esempio la centralità del nodo. Evidenzia le connessioni fra i nodi, ha complessità $O(n)$.

Radial: Molto simile a Circular, i nodi sono posti su una circoferenza. I nodi simili fra loro verranno però raggruppati su assi, che si irradieranno verso l'esterno a partire dalla circonferenza. [5]

Gephi presenta un buon numero di layout su cui lavorare, e quelli sopra citati sono solo alcuni.
E' fondamentale capire come si comportano i layout a seconda del tipo di grafo, e di conseguenza scegliere il layout più appropriato che ci restituisca il grafo nella forma che ci interessa. Lo stesso grafo lavorato, ad esempio, con Force Atlas e con Radial assumerà una forma molto diversa, di seguito:

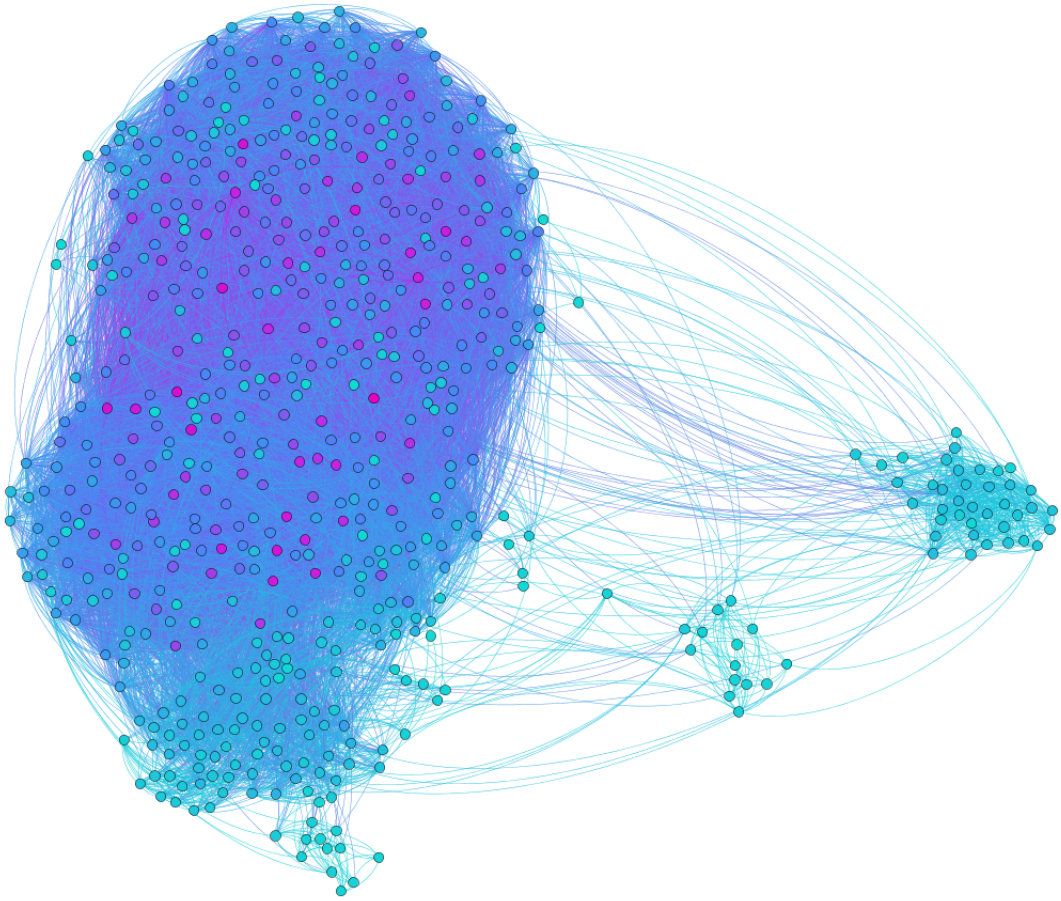


Figura 3.3: Force Atlas

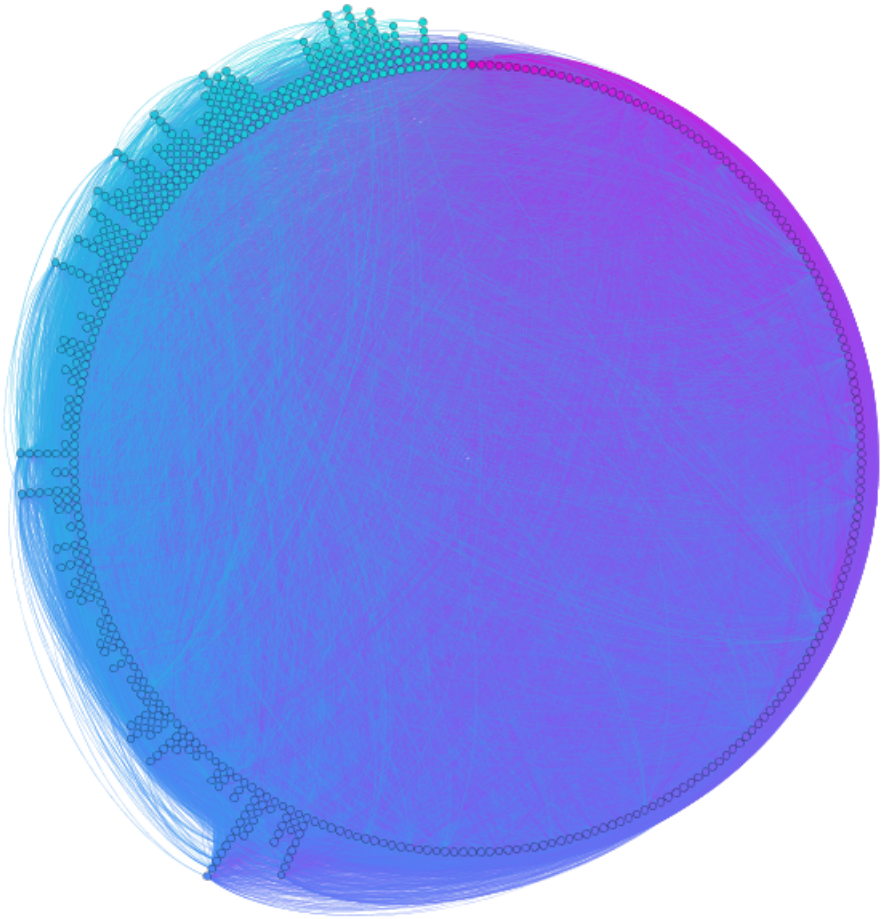


Figura 3.4: Radial

3.2.3 Statistics

Alcuni nodi sono ovviamente più importanti degli altri. Per calcolare il loro peso all'interno del grafo Gephi mette a disposizione alcune funzioni:

Pagerank: è un algoritmo iterativo che misura l'importanza di ogni nodo all'interno della rete. Pensando ai nodi come a pagine web, la funzione assegna a ciascun nodo la probabilità di ricomparire dopo molti click. I pagerank delle pagina saranno i valori degli autovettori che hanno autovalori più alti nella matrice di adiacenza. La matrice è normalizzata in modo tale che le colonne si sommino a 1.

HITS: calcola due punteggi: "hub" e "autorità". Hub sarà calcolato contando il numero di connessioni in uscita, più il valore hub sarà alto, più il nodo sarà centrale, un fulcro di smistamento. L'autorità misurerà l'importanza di un nodo sulla base della posizione e dei collegamenti, se un nodo è collegato con hub importanti la sua autorità avrà valore maggiore. E' un algoritmo iterativo, e per ogni iterazione:

- Aggiorna il valore autorità di ogni nodo, calcolandolo in base ai collegamenti con nodi hub. Il valore autorità sarà la somma dei valori hub a cui il nostro nodo è collegato. Ovviamente, più gli hub saranno importanti più i loro valori saranno alti, e il nostro nodo avrà autorità più alta;
- Aggiorna il valore hub per ogni nodo, calcolandolo in base alle autorità. Il valore hub sarà la somma dei valori autorità dei nodi con cui è collegato;
- Normalizza i punteggi hub e autorità;
- Ripete le operazioni fin quando non diventano costanti;

Average Lenght Path: in un grafo, la distanza γ fra due nodi *u,v* è definita come il minor numero di archi necessario per raggiungere *v* partendo da *u*. I valori saranno uguali in un grafo non orientato, mentre potrebbero essere diversi in un grafo orientato. In un Network, il valore medio del percorso fra tutti i nodi può essere interpretato come misura dello "Small World effect", anche chiamato "teoria dei sei gradi di separazione".

Network Diameter: trova la γ più alta in tutto il grafo, per misurare il diametro del network.

Node Betweenness Centrality: controlla quante volte un nodo appare nel calcolo di "Average Shortest Path", ovvero quanto spesso si incontra un nodo calcolando i cammini minimi fra altri.

$$CB(v) = \sum_{u,w \in N, u \neq v \neq w} \frac{\sigma_{u,v}(v)}{\sigma_{u,v}},$$

dove $\sigma_{u,v}(v)$ è il numero dei cammini minimi da s a t che passano per il nodo v, e $\sigma_{u,v}$ è il numero dei cammini minimi da s a t.

Modularity: i Network hanno dimostrato di essere separati in gruppi logici in cui i nodi sono strettamente collegati l'uno all'altro, ma debolmente legati con i nodi di fuori del loro cluster. La modularità di Newman è attualmente quella più usata per stabilire quanto un Network è modulare. Data una partizione P in un Network G, definiamo la modularità come:

$$Q(P, G =< N, E >) \sum_{C_i \in P} \frac{l(C_i)}{|E|} - \left(\frac{d(C_i)}{2 * |E|}\right)^2$$

dove C_i è un modulo della partizione, $l(C_i)$ è il numero di archi che collegano i nodi dentro C_i e $d(C_i)$ è il grado del modulo. Il termine sottratto è il quadrato della percentuale di archi che terminano in C_i. [6]

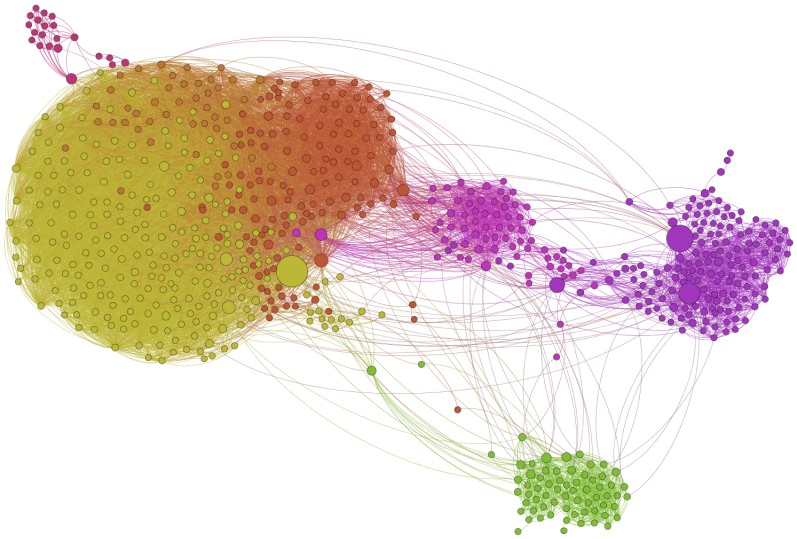


Figura 3.5: Notiamo i gruppi di nodi di colori diversi, dove ogni colore rappresenta un modulo, e le diverse dimensioni dei nodi grazie alla funzione Avg. Lenght Path

Capitolo 4

I Social Network e i grafi

4.1 L'analisi dei Social Network

La Social Network Analysis è lo studio e l'analisi delle reti sociali. L'idea è quella di tradurre le relazioni sociali in termini di teoria delle reti, dove i nodi rappresentano persone e i collegamenti le loro relazioni (amicizia, parentela, lavoro). Possiamo pensarla, piuttosto che a uno studio teorico, come un modo vero e proprio di investigare all'interno di una rete. I soggetti principali all'interno di un network non sono gli individui stessi, ma le relazioni fra gli individui e le loro azioni, il loro comportamento. Esistono due forme di analisi dei social network: la prima dove si investiga su un solo individuo e la sua rete, e la seconda dove si studia un network completo, cercando di analizzare ogni relazione fra i componenti del network. La famosa teoria dei "Sei gradi di separazione" si basa proprio sull'analisi dei social network, dove un network di persone, nonostante l'ampia estensione, ha comunque una distanza media fra i nodi molto piccola, assicurando al massimo 6 passaggi per arrivare da un nodo a un altro qualunque (Small World Network). [7] Sono numerosi i casi dove i grafi e la Social Network Analysis sono stati di grande utilità, riassumendo in un disegno quello che normalmente sarebbe stato solo un elenco di voci. In fondo, se possiamo dire che un'immagine vale più di mille parole, un grafo vale tranquillamente più di mille righe. Citerò qui qualche esempio di analisi interessante fatta sui Social Network.

4.2 Facebook

4.2.1 Netvizz

Netvizz è un'applicazione Facebook: dopo aver chiesto il permesso per accedere al profilo restituirà una schermata molto semplice, apparentemente un form, dove poter selezionare alcune voci. Si possono includere più o meno informazioni per la creazione del proprio grafo sociale, come il sesso, l'età, il numero di post in bacheca, tutti fattori che determineranno la completezza e l'accuratezza del grafo. Una volta estrapolati i dati, viene restituito un file *.gdf*. Il grafo creato conterrà tutti i nostri amici e un certo numero di informazioni che abbiamo scelto o no di includere, e presenterà le relazioni fra di loro. Facendo un test di prova con il mio profilo Facebook, su una base di 400 amici, Netvizz ha impiegato circa due minuti per produrre il file. Provando ad importarlo in Gephi (come un grafo orientato), il primo risultato è quello di un grafo molto casuale senza alcun ordinamento.

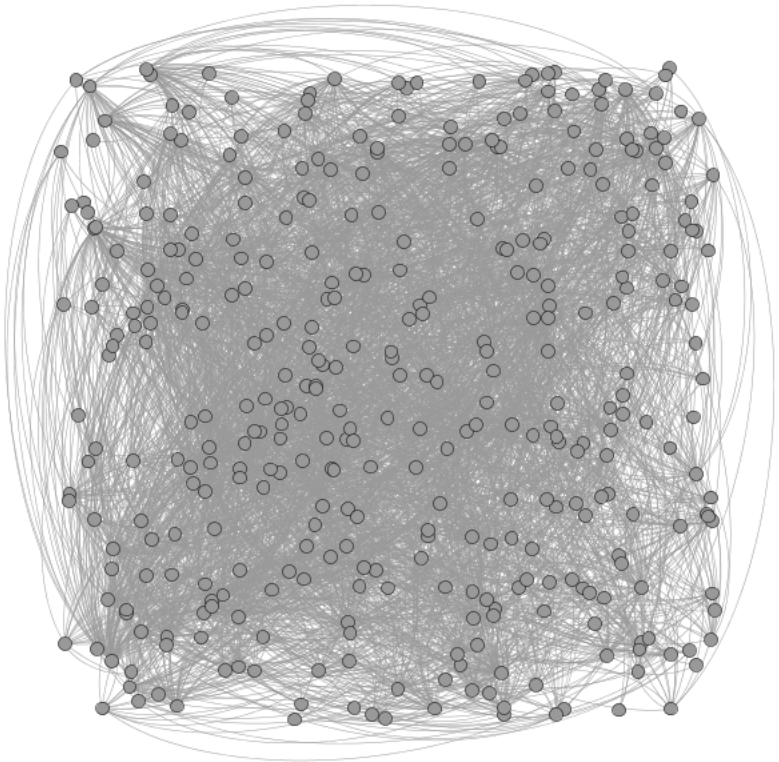


Figura 4.1: Il file .gdf importato

Lanciando il layout *Force Atlas* notiamo un cambiamento della forma del grafo. I nodi più vicini e connessi fra loro si raggrupperanno in agglomerati, mentre quelli meno

connessi si allontaneranno per formare agglomerati minori con altri nodi, o semplicemente verranno isolati e posti verso la periferia del grafo perché sconnessi.

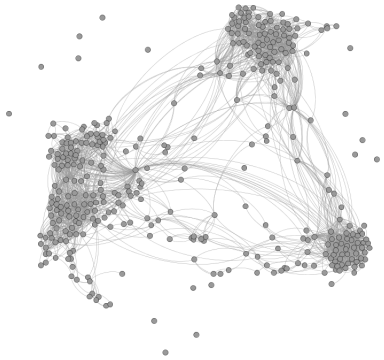


Figura 4.2: Il grafo dopo la lavorazione con *Force Atlas*

Per colorare un po' il grafo e dare uno sguardo all'importanza dei singoli nodi, nella tabella "Ranking" in alto a sinistra selezioniamo la voce "Degree", la quale colorerà del colore che preferiamo i nodi con più connessioni. In questo caso, i nodi fucsia saranno quelli più connessi con altri, mentre verso l'azzurro saranno meno importanti.

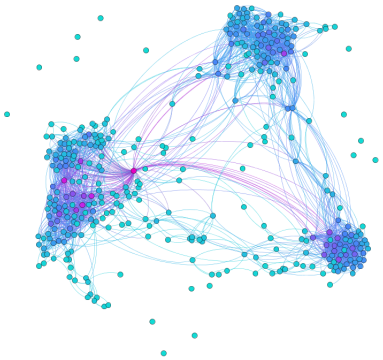


Figura 4.3: Applicando Degree

Nella sezione “Statistics” ora lanciamo la funzione *Average Path Lenght*. Clicchiamo su Directed, perché il nostro grafo è orientato, e aspettiamo il risultato. Nel mio caso, il diametro del grafo risulta essere di 11, ovvero i gradi di separazione massimi fra due amici che non si conoscono sono 11. Naturalmente io non sono inclusa nel grafo, altrimenti il massimo grado di separazione sarebbe 2. Il percorso medio fra due punti del grafo risulta essere 2.9. Per visualizzare il risultato sul grafo, selezioniamo nella tab “Ranking” il parametro “Betweenness Centrality” sulla dimensione del nodo. Il risultato sarà l’aggiornamento delle dimensioni dei nodi, dove i più grandi saranno i nodi più di passaggio, ovvero quelli dai quali si passa più spesso nel percorso minimo da un nodo all’altro.

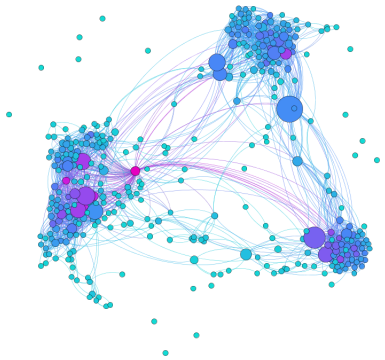


Figura 4.4: Applicando Avg. Path Length

Ora i nodi però appaiono sovrapposti. Per evitare che si confondano fra loro, rilancio *Force Atlas* aggiungendo una spunta sul parametro “Adjust by sizes”. I nodi si separeranno e i cluster saranno più chiari. Come ultima cosa lancio nel pannello “Statistics” la funzione *Modularity* , la quale identificherà i moduli all’interno del grafo e li colorerà diversamente gli uni dagli altri. [8]

Nel mio caso, è curioso notare come il modulo di nodi gialli siano tutti gli amici che ho conosciuto a Bologna, il modulo arancione tutte le persone del mio paese di origine (Lugo), quelli blu tutti i fotografi, e quelli verdi tutti i miei colleghi universitari. Una volta conclusa la lavorazione del grafo è possibile esportarla dalla tab “Preview” e aggiungere, eventualmente, etichette (ovvero nomi dei nodi) o modificare altri parametri.

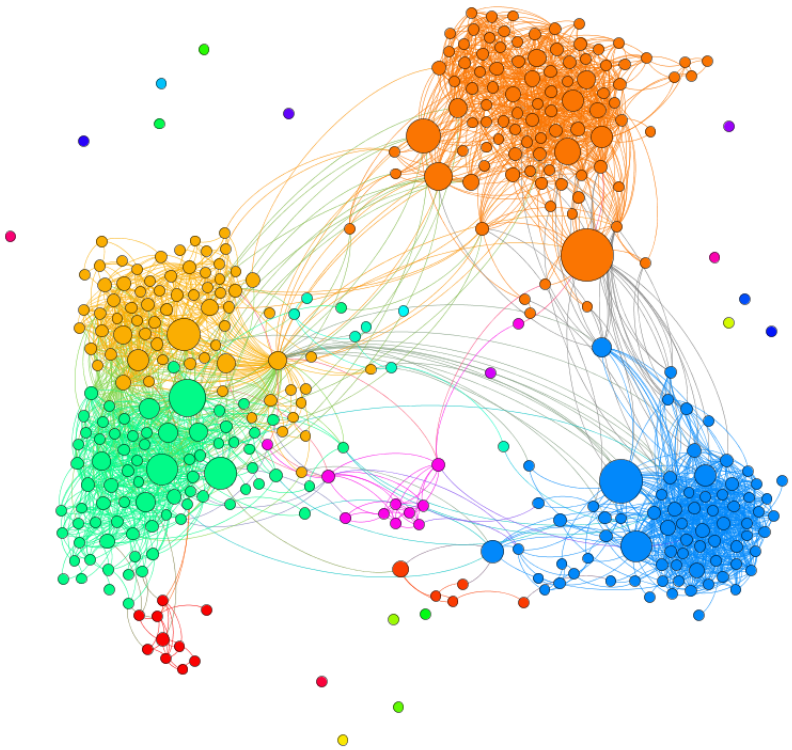


Figura 4.5: Modularity

4.3 Twitter

4.3.1 L'umore americano tramite Twitter

Per oltre tre anni, i ricercatori di Northeastern e Harvard hanno esaminato tweets (da settembre 2006 ad agosto 2009) per rappresentare l'emotività degli Americani e i loro cambiamenti d'umore durante la giornata.
Sapendo di poter contare sull'emotività delle persone e la loro tendenza a condividere i pensieri in 140 caratteri, la ricerca si è concentrata su alcune parole chiave che indicavano l'umore di una persona, combinati con dati forniti dal Census e dalle mappe Google. Nell'infografica, il verde corrisponde ad uno stato d'animo felice e il rosso alla negatività. L'area di ogni Stato viene scalata a seconda del numero di tweet raccolti. Si nota anche che in Californa e Florida la media di felicità è più alta, grazie anche al bel tempo. [9]

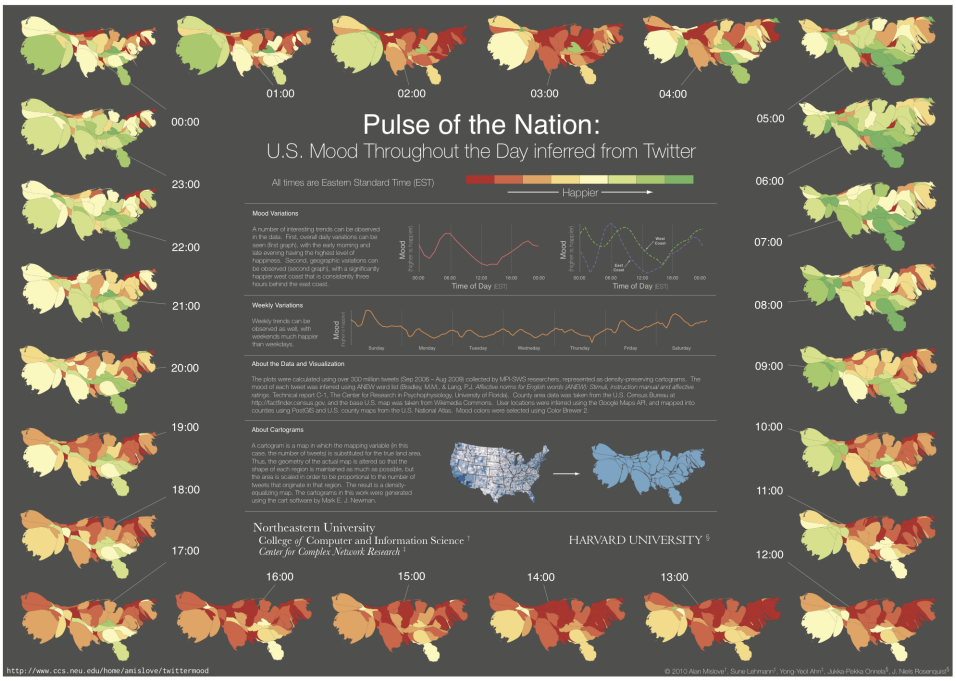


Figura 4.6: L'infografica sugli stati d'animo americani

4.3.2 L'uso di Twitter durante il terremoto

E' noto che Twitter sia il social network più immediato, dove le informazioni circolano più velocemente. Uno degli aspetti positivi di questo social è la sua restrizione e regola principale: far circolare contenuti in meno di 140 caratteri. Gli aggiornamenti degli utenti sono quindi sempre brevi e diretti, liberi da fronzoli ed efficaci.

Durante il terremoto in Emilia del 20 maggio le notizie riguardanti le prime scosse sono partite da twitter, e si sono propagate su quotiani online sono qualche ora dopo. La principale fonte di informazioni è stata, come ogni volta, la massa di utenti che ha aggiornato il proprio profilo scrivendo qualcosa riguardo al terremoto, facendolo sapere alle persone che non erano sul luogo della scossa. Anche se IngvTerremoti ha pubblicato la magnitudo esatta della scossa principale solo mezz'ora, è stato comunque uno fra gli account più retwittati e di fondamentale importanza. Strano ma vero, i VIP sono diventati fulcro del traffico twitter grazie al loro enorme numero di followers, trasformandosi in portavoce improvvisati a distanza. Il progetto di ricerca "Relazioni sociali ed identità in Rete: vissuti e narrazioni degli italiani nei siti di social network" si propone di indagare le forme di costruzione e ridefinizione delle relazioni sociali e dell'identità degli italiani in Rete, utilizzando i siti di social network come luogo di osservazione privilegiato. Grazie al loro monitoraggio attivo su alcune hashtag, compresa "terremoto", durante le prime ore dopo le scosse hanno potuto impacchettare e studiare il flusso di dati che ha attraversato twitter.

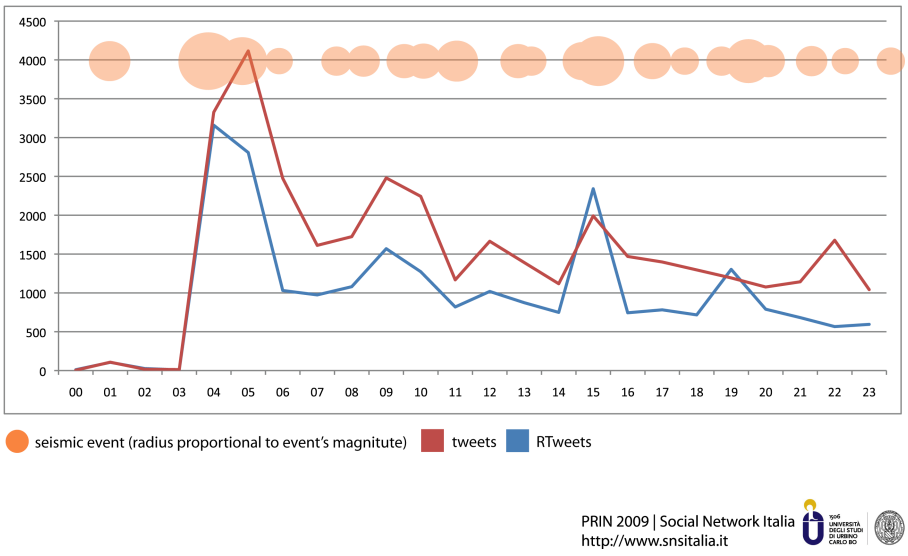


Figura 4.7: La curva di tweet rispetto agli eventi sismici

Da questa prima visualizzazione è evidente un chiaro picco delle attività Twitter in corrispondenza delle scosse maggiori (sia la principale delle 4.03 che quelle di assestamento susseguitesi tutto il giorno). Possiamo però notare qualche differenza fra il picco tra le 4 e le 6 e quello fra le 15 e le 17. Nel primo caso, la produzione di Tweet originali è di molto maggiore rispetto a quella di ReTweet, mentre tra le 15 e le 17 – in occasione di una seconda forte scossa – questo rapporto si inverte e la produzione di ReTweet supera

la produzione di Tweet originali.

Se nella prima TagCloud la centralità è – oltre che per la parola scossa – per tutta una serie di riferimenti geografici volti a definirne l'area e l'intensità (epicentro, magnitudo), nella seconda scossa è possibile osservare la comparsa di riferimenti alla protezione civile, i numeri di telefono utili e le espressioni di call for actions come "contattare", "segnalare", e "ritwittate". [10]

Figura 4.8: TagCloud del picco fra le 4 e le 6

Figura 4.9: TagCloud del picco fra le 15 e le 17

Dopo ogni evento di crisi, la fase più importante è quella di coordinamento ed organizzazione dei soccorsi. Dallo studio di SNSItalia, il caso del terremoto del 20 Maggio 2012 mette in evidenza interessanti dinamiche auto-organizzative, ma anche l'assenza di soggetti istituzionali in grado di offrire informazioni certe, farsi punto di riferimento per le richieste e coordinare le attività spontanee. La conclusione del team è che in Italia questo ruolo sia svolto da account popolari come quelli delle celebrities e altri user molto seguiti, in assenza di account istituzionali. Probabilmente, la creazione di account istituzionali inseriti nel tessuto comunicativo dei social media potrebbe rivelarsi una essenziale forma di informazioni in caso di eventi di crisi. [10]

Capitolo 5

Il caso *#terremoto*

5.1 Obiettivi

L'obiettivo del mio studio è quello di mappare e visualizzare la grande quantità di tweet che è circolata durante le prime scosse di terremoto. Raccogliendo più tweet possibili li posizionerò su una mappa per focalizzare l'attenzione sulla geolocalizzazione dei tweet, e verificare se la maggior parte di essi si è generata proprio intorno all'epicentro. Realizzerò anche un video che mostra l'acquisizione in tempo reale dei trends, per mostrare come si muove il traffico twitter e identificare quelli che possono essere, di volta in volta, gli user più influenti e come si concentra il traffico intorno ai loro tweet. Visualizzando graficamente le informazioni darò forma a tabelle e le studierò come reti di persone, metterò i tweet in relazione fra di loro e studierò il loro andamento nel tempo.

5.2 Introduzione a Twitter

Nato nel 2006 a San Francisco, Twitter è un social network che fornisce agli utenti una pagina profilo da aggiornare di volta in volta con messaggi di testo lunghi al massimo 140 caratteri. E' una piattaforma gratuita di microblogging, costruita interamente su architettura Open Source. Gli aggiornamenti possono essere effettuati tramite il sito stesso, via SMS, con programmi di messaggistica istantanea, posta elettronica, oppure tramite varie applicazioni basate sulle API di Twitter. Il nome "Twitter" deriva dal verbo inglese to tweet: "cinguettare". Gli aggiornamenti sono mostrati istantaneamente nella pagina di profilo dell'utente e comunicati agli utenti che si sono registrati per riceverli. È anche possibile limitare la visibilità dei propri messaggi oppure renderli visibili a chiunque. Il valore di questo social network è stato stimato intorno agli 8,4 miliardi di dollari e il 22 febbraio 2012 ha raggiunto i 500 milioni di utenti attivi che fanno accesso almeno una volta al mese.

Il servizio è diventato estremamente popolare grazie alla semplicità ed immediatezza di utilizzo. Esistono diversi esempi in cui Twitter è stato usato dagli utenti per diffondere notizie, come strumento di giornalismo partecipativo. Ad esempio, nel caso del terremoto in Abruzzo del 6 aprile 2009 gli utenti Twitter hanno segnalato la notizia prima dei media tradizionali, come anche è successo durante il terremoto dell'Emilia il 20 maggio 2012. L'insieme degli "status message" pubblicati su Twitter dagli utenti costituisce un'enorme quantità di materiale, che può essere utilizzata anche dalle aziende: Vodafone (*@VodafoneIT*), Trenitalia (*@lefrecce*), Poste Italiane(*@PosteSpedizioni*), ad esempio hanno aperto un canale di comunicazione con i propri clienti su Twitter per assistenza e per raccogliere feedback. [11]

5.2.1 Interfaccia

Twitter ruota intorno al principio dei seguaci (followers). Quando si sceglie di seguire un altro utente di Twitter, i tweets di tale utente vengono visualizzati in ordine cronologico inverso, sulla home page di Twitter. Se seguite 20 persone, si vedrà una miscela di tweets scorrere la pagina. I temi di attualità (Temi di Tendenza, o Trending Topics), cioè le frasi più comuni che compaiono nel messaggio sono ricercabili all'interno dell'interfaccia Twitter e rappresentano gli argomenti più twittati e più in voga del momento. Ogni aggiornamento pubblico inviato a Twitter da qualsiasi parte del mondo può essere immediatamente indicizzato e utilizzato per la ricerca in tempo reale, diventando un motore di ricerca per trovare proprio ciò che sta accadendo in questo momento. [11]

5.2.2 Hashtag

L'hashtag è una parola o una frase preceduta dal simbolo cancelletto (#) con più parole concatenate, come ad esempio: "#lunedì è la giornata peggiore della settimana, avrei proprio voglia di #vacanze".

In questo modo una persona può cercare il termine #vacanze e la parola etichettata apparirà nei risultati di ricerca. Questi hashtag appaiono anche in un certo numero di siti web di termini più trattati (trending topics), tra cui la homepage di Twitter. Gli hashtag di Twitter possono essere utilizzati per seguire una discussione tra più persone, incoraggiando altre persone a partecipare. Un fenomeno specifico degli ecosistemi Twitter sono i micro-meme, che sono le questioni emergenti che vengono seguite con un hashtag, ampiamente usato per un paio di giorni e che poi sparisce. Nel 2012 sono stati introdotti i trending topics localizzati che permettono la visualizzazione degli hashtag più popolari di ogni Stato. [11]

5.3 L'acquisizione dati

Uno dei primi problemi che ho incontrato per svolgere la mia ricerca è stato quello dell'acquisizione dati. Cercando normalmente l'hashtag #terremoto all'interno del sito Twitter.com i risultati erano ovviamente tutti molto recenti, e scorrendo indietro fra le pagine c'era un limite di circa una settimana. Controllando i parametri per la ricerca avanzata "since" e "until" i tweet restituiti non andavano comunque più indietro di una settimana, e fare una ricerca riguardante mesi prima non era quindi possibile. Controllando sulle API di Twitter (http://dev.twitter.com) ho avuto la conferma: le API non permettono di ricercare fra tweet più vecchi di una settimana.

I tweet del 20 maggio erano quindi inarrivabili attraverso qualsiasi software e l'unico modo per reperirli era contattare qualcuno che lo avesse fatto in tempo reale, quindi al momento delle prime scosse, tramite un qualche sistema di monitoraggio.

Il sito SNSItalia (http://www.snsitalia.wordpress.com) di cui ho parlato in precedenza era riuscito a reperire i dati al momento della scossa, avendo attivo un sistema di monitoraggio su varie hashtag per attività di ricerca, fra cui casualmente anche #terremoto. Sono riuscita quindi ad accedere al loro archivio di dati per esportare due campioni di tweet.

Dall'archivio è possibile specificare un range molto ampio per quanto riguarda dati e numero di tweet. I dati sono stati raccolti a partire dal 5 maggio 2012 e perciò ricadono perfettamente nel periodo interessato, e si possono esportare da 10 fino a 10.000.000 tweet. La mia scelta è stata quella di prelevare due campioni: il primo da 10.000 tweet dal 19 al 20 maggio, e il secondo da 10 milioni di tweet dal 20 al 22 maggio. Il primo file mi servirà per lavorare sul traffico vero e proprio dei tweet, sulle relazioni fra gli user e i tweet più retwittati (quindi quelli più popolari durante le scosse). Il secondo campione mi servirà per geolocalizzare più tweet possibili. Nonostante avessi deciso, per il secondo campione, di prendere fino a 10 milioni di tweet il numero è sceso a 282.163, ovvero il traffico totale di tweet per quanto riguarda quei giorni. Il file esportato è un *.csv* che andrà opportunamente elaborato per poter guadagnare utilità.

5.4 Formattazione e conversione dei dati

Una volta ottenuti i due file, bisogna convertirli e formattarli nel modo appropriato. I file ci vengono forniti come *.csv* (comma separated values), dove appunto i campi sono delimitati da virgole. All'interno del file i campi che troviamo sono:

```
text, to_user_id, from_user, id, from_user_id,
iso_language_code, source, profile_image_url,
geo_type, geo_coordinates_0,
```

`geo_coordinates_1, created_at, time.`

Quelli che mi interessano maggiormente sono quelli che riguardano il contenuto dei tweet (text), lo user di provenienza, le coordinate (geo_coordinates_0, geo_coordinates_1) e, nel caso in cui si tratti di una risposta, il campo to_user_id. Il file risulta abbastanza pulito, a parte alcuni campi tranquillamente eliminabili come quello riferito all'immagine profilo dello user o del tipo di codifica utilizzato per mandare il tweet, tuttavia la mia scelta è stata quella di convertire i delimitatori da virgole a pipe, perché nel contenuto dei tweet le virgole sono caratteri di punteggiatura spesso utilizzati, e una volta importato in Gephi il file si sarebbe spezzato in più parti generando una grande quantità di dati falsati.

Perciò ho convertito i delimitatori da virgole a pipe tramite *CsvEditor*, un editor scritto in Java, ottenendo un file più adatto all'importazione in Gephi.

5.4.1 R

Il file che ho importato in Gephi è sprovvisto di archi: se visualizzato, si presenta semplicemente come un insieme di punti sconnessi fra loro.

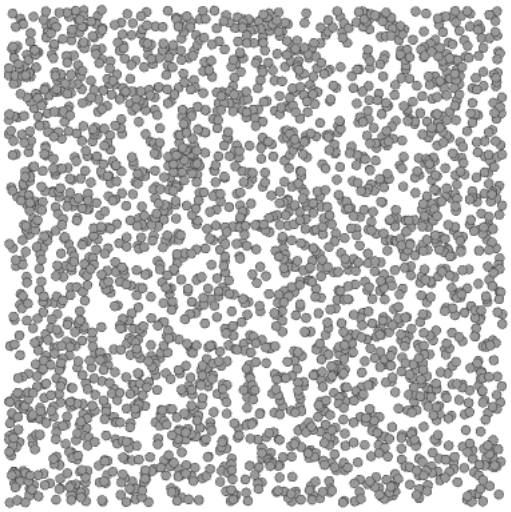


Figura 5.1: Prima importazione del file in Gephi

Naturalmente è impensabile connettere 10.000 nodi fra di loro manualmente, di conseguenza ho cercato un modo per collegare fra loro i nodi e dare una forma al grafo. Due relazioni esaminabili fra i tweet sono i "retweet", ovvero i tweet che sono stati prodotti in origine da uno user ma condivisi da altri, e le "mentions", ovvero i tweet indirizzati

direttamente a uno user tramite l'apposita chiocciola apportata vicino al nickname dell'utente con cui vogliamo comunicare (es: @Owlage).

Tramite queste relazioni i tweet verranno connessi fra loro per formare una ragnatela e riuscire ad interagire fra loro. "R" è un ambiente di sviluppo specifico per l'analisi statistica dei dati che utilizza un linguaggio di programmazione derivato e in larga parte compatibile con S. L'utilizzo di R si è rivelato fondamentale nel nostro caso: tramite l'interfaccia a riga di comando possiamo manipolare il file *.csv* ed esportarlo in un *.graphml*, comprensivo quindi di connessioni fra nodi secondo le relazioni che preferiamo, ed essere così pronto per essere importato in Gephi. Lo script che ho utilizzato per la conversione del file è il seguente [12]:

```
1
2 library(igraph);
3
4 tweets <- read.csv("tweets.csv", head=T, sep="|", quote="",
    fileEncoding="UTF-8");
5 print(paste("Read ", length(tweets$text), " tweets.", sep=""));
6
7 ats <- grep("^\\.?@[a-z0-9_]{1,15}", tolower(tweets$text), perl=T,
    value=T);
8 at.sender <-
    tolower(as.character(tweets$from_user[grep("^\\.?@[a-z0-9_]{1,15}",
    tolower(tweets$text), perl=T)]));
9 at.receiver <- gsub("^\\.?@([a-z0-9_]{1,15})[^a-z0-9_]+.*$", "\\1", ats,
    perl=T);
10 print(paste(length(ats), " @-messages from ",
    length(unique(at.sender)), " senders and ",
    length(unique(at.receiver)), " receivers.", sep=""));
11
12 rts <- grep("^rt @[a-z0-9_]{1,15}", tolower(tweets$text), perl=T,
    value=T);
13 rt.sender <- tolower(as.character(tweets$from_user[grep("^rt
    @[a-z0-9_]{1,15}", tolower(tweets$text), perl=T)]));
14 rt.receiver <- gsub("^rt @([a-z0-9_]{1,15})[^a-z0-9_]+.*$", "\\1", rts,
    perl=T);
15 print(paste(length(rts), " RTs from ", length(unique(rt.sender)), "
    senders and ", length(unique(rt.receiver)), " receivers.", sep=""));
16
17 at.sender[at.sender==""] <- "<NA>";
18 at.receiver[at.receiver==""] <- "<NA>";
19 rt.sender[rt.sender==""] <- "<NA>";
20 rt.receiver[rt.receiver==""] <- "<NA>";
21
22 ats.df <- data.frame(at.sender, at.receiver);
23 rts.df <- data.frame(rt.sender, rt.receiver);
24
```

```
25 ats.g <- graph.data.frame(ats.df, directed=T);
26 rts.g <- graph.data.frame(rts.df, directed=T);
27
28 print("Write sender -> receiver table to GraphML file...");
29 write.graph(ats.g, file="ats.graphml", format="graphml");
30 write.graph(rts.g, file="rts.graphml", format="graphml");
```

Lo script procede leggendo il *.csv* tenendo conto delle pipe come separatori, cerca sia le chiocciole che i retweet e salva i valori che verranno poi esportati come due file separati, un file *ats.graphml* e uno *rts.graphml*. I file sono separati per conferire maggiore chiarezza ai grafi finali e per non mescolare fra loro relazioni di mention e di retweet.

5.5 Manipolazione in Gephi

Una volta ottenuti i due file possiamo aprirli in Gephi da "File - Open". La finestra apparsa comunicherà una serie di informazioni: il numero di nodi, il numero di archi, e chiederà se vogliamo importare il grafo come orientato o come non orientato. Importerò il grafo come orientato.

Nel grafo, oltre ai nodi, appariranno anche gli archi. In questo caso, lavorando sul file *rts.graphml* gli archi rappresenteranno la condivisione di un tweet fra più utenti, e gli utenti saranno collegati fra loro solo se uno ha retwittato l'altro o viceversa.
Come prima cosa lanciamo un layout per individuare i cluster. *Force Atlas*, benché efficace, lavora con estrema lentezza (avendo complessità $O(n^2)$), ed è effettivamente un po' al limite per il nostro grafo. *Force Atlas 2* è più veloce, ma i cluster restano comunque molto vicini fra loro e non si riesce a notare un vero e proprio evolversi del grafo, piuttosto uno "sparpagliarsi" dei piccoli agglomerati di nodi senza produrre un risultato efficace. Un netto cambiamento avviene una volta lanciato *Open Ord* con i parametri nella figura sottostante. Il layout procede e termina autonomamente, per presentarci un grafo ben diviso che si avvicina molto a una rappresentazione reale del traffico twitter.

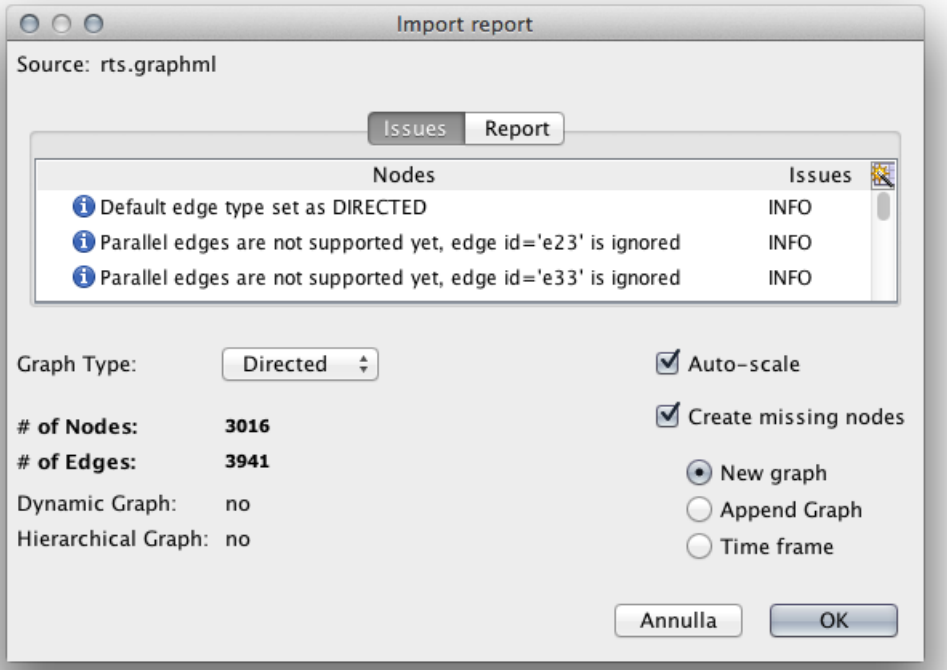


Figura 5.2: Finestra di dialogo

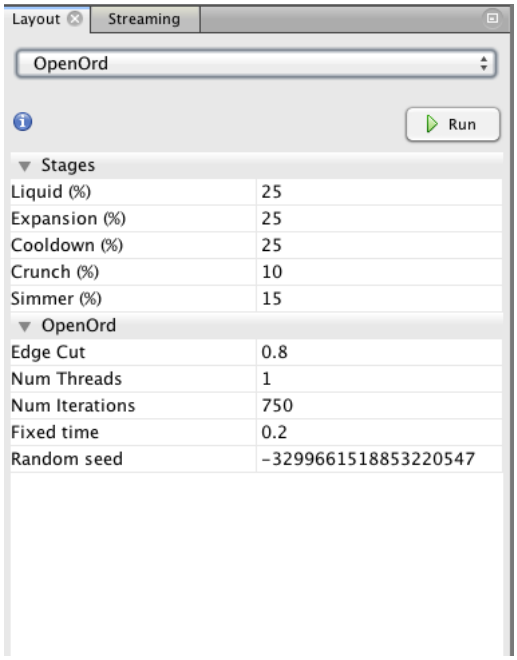


Figura 5.3: Parametri di OpenOrd

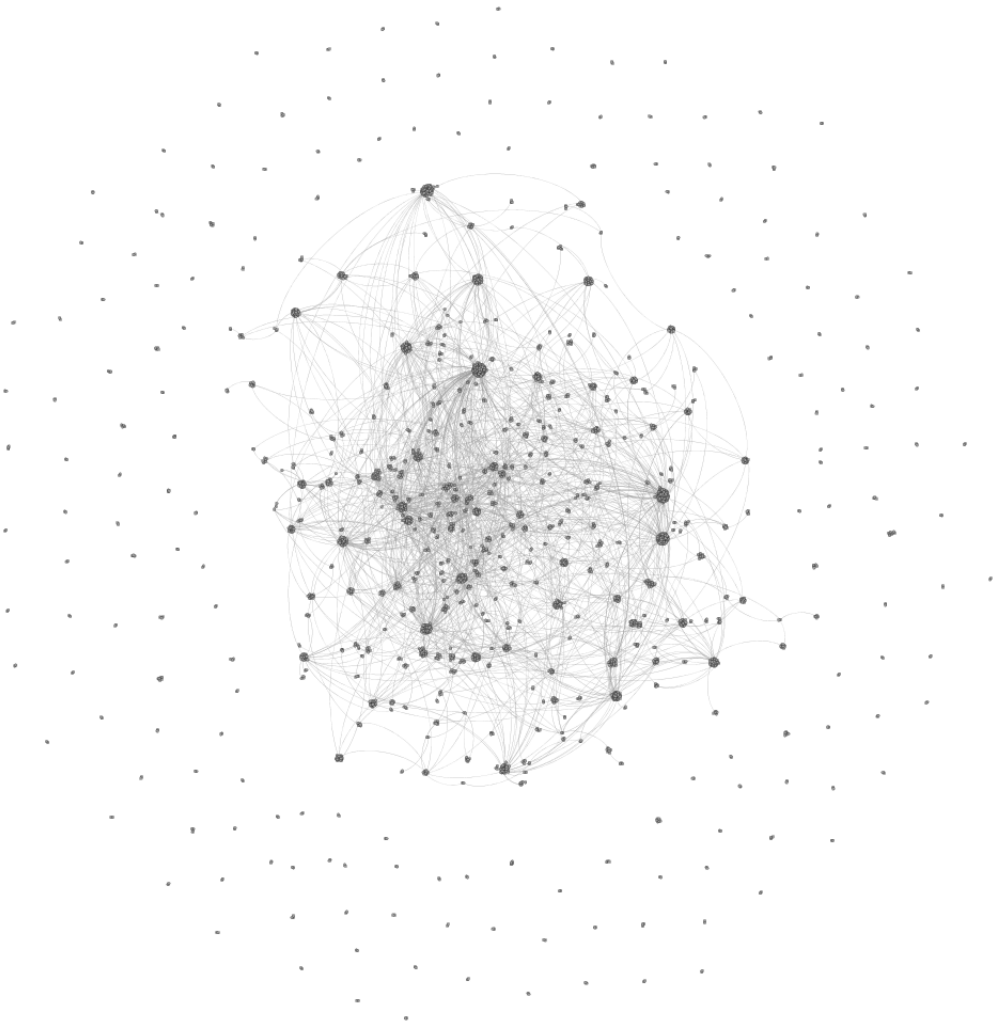


Figura 5.4: OpenOrd

Al centro si identificano dei piccoli agglomerati che rappresentano uno user e tutti quelli che lo hanno retwittato. Nella periferia del grafo piccoli agglomerati di minimo due nodi ciascuno, ovvero uno user e un solo altro user che ha retwittato. Ovviamente non si trovano nodi singoli sparsi, perché la relazione di retweet è almeno fra due nodi.

Il passo successivo consiste nel lanciare, dalla tabella "Statistics" la funzione *"Average Path Lenght"*. La funzione analizzerà il nostro grafo e ne ricaverà informazioni utili come: il diametro del grafo (ovvero la distanza massima fra due nodi non connessi direttamente), il numero medio di archi fra un nodo e l'altro, e ci permetterà di dare ad ogni nodo un peso diverso a seconda della sua centralità all'interno del grafo. Selezionando nella tab "Ranking" l'opzione "Betweeness Centrality", i nodi che si incontrano più spesso nel calcolare i cammini minimi aumenteranno di peso, e verranno riconosciuti appunto come più centrali. Scegliendo una dimensione da 10 a 50 punti per i nodi, una volta applicatoquelli più significativi appariranno più grandi degli altri. E' necessario sistemare i piccoli agglomerati per evitare che i nodi si sovrappongano gli uni agli altri lanciando *Force Atlas* per pochi secondi, inserendo la spunta alla voce "Adjust by Sizes". I nodi si allontaneranno leggermente gli uni dagli altri lasciando ad ognuno il giusto spazio.

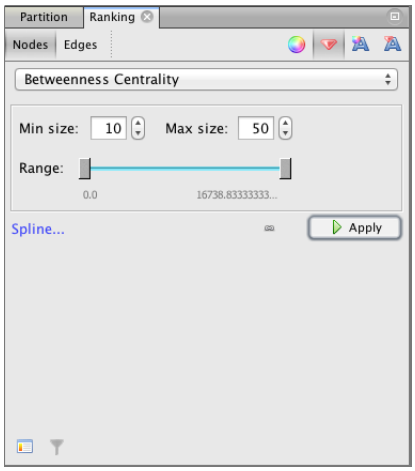


Figura 5.5: Ranking

Una volta fatto, 'Modularity" dalla sezione "Statistics" colorerà i cluster.

Non resta che inserire le etichette vicino ad ogni nodo per identificare i nickname degli user più retwittati. Il file finale sarà simile sia con le relazioni di retweets sia con le mentions, ma nel caso dei retweets i dati saranno molti di più e comunque più significativi, perché rappresentano la diffusione di una notizia.

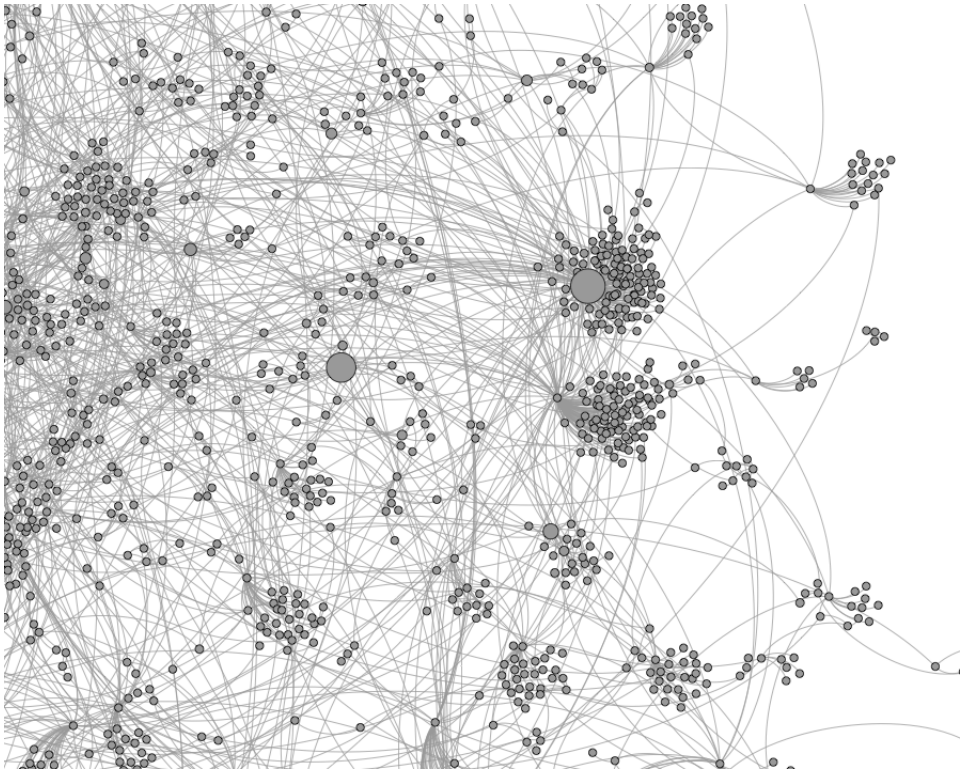


Figura 5.6: Centrality

5.6 Esportazione del grafo

Il grafo può essere esportato sia in formato *.graphml*, *.gephi*, *.csv* oppure anche come *.pdf*, *.swg*, *.png* se ci interessa solo il grafo finale che non andremo più a manipolare ma solo a visualizzare.

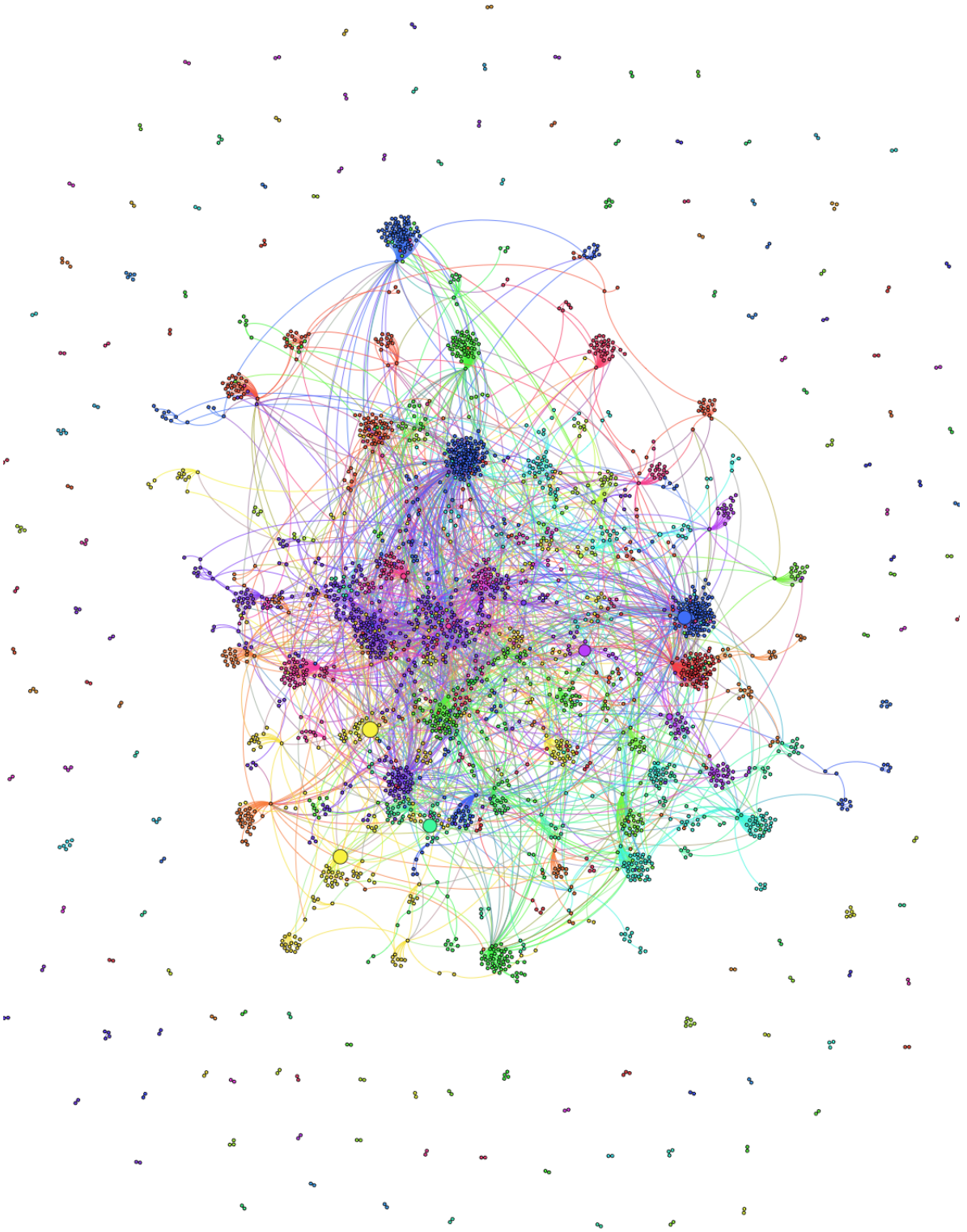


Figura 5.7: Modularity

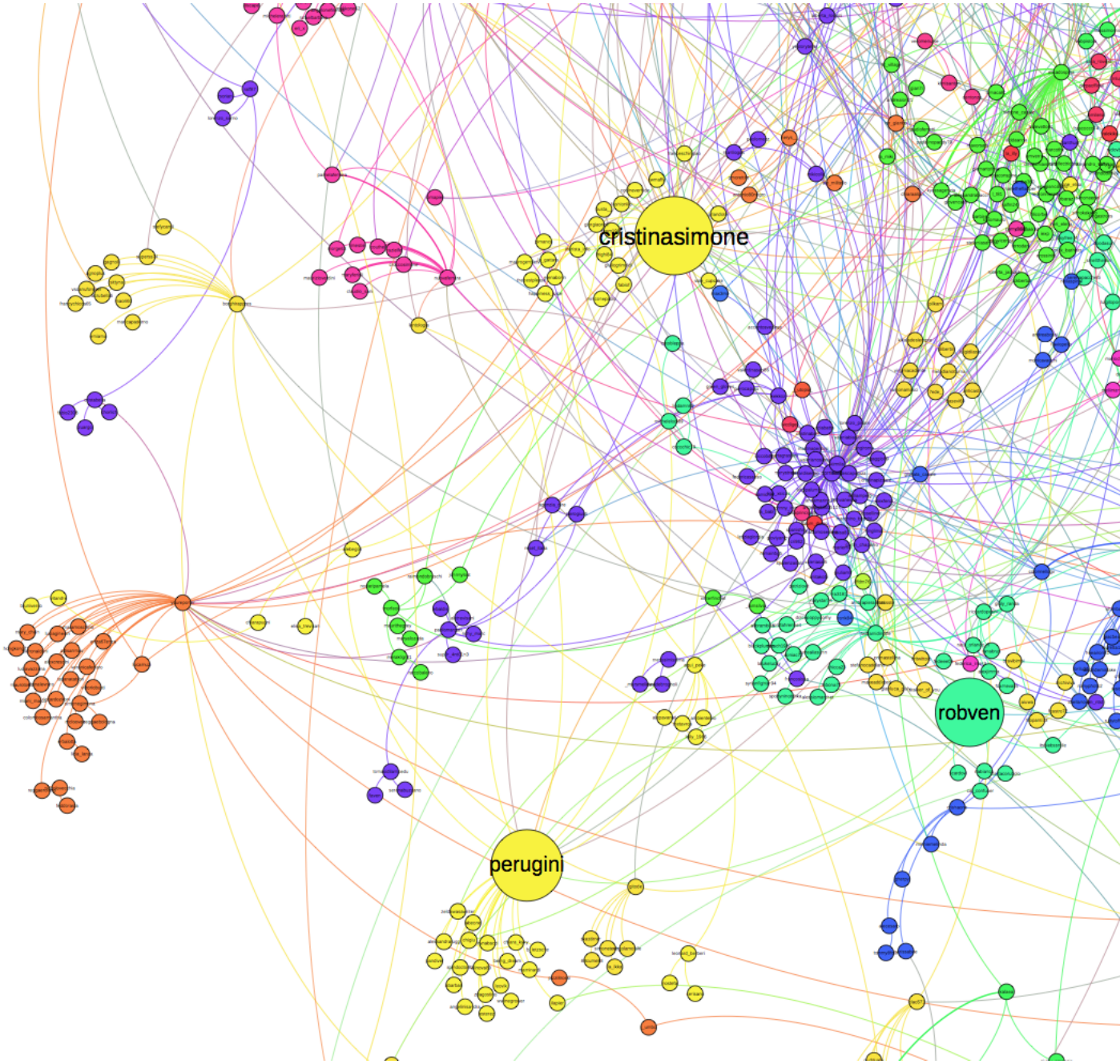


Figura 5.8: Dettaglio del grafo completo

5.7 I Tweet geolocalizzati

5.7.1 La geolocalizzazione in Gephi

GeoLayout è un layout molto interessante, incluso solo recentemente nei layout principali di Gephi, che permette il posizionamento dei nodi tramite due coordinate decimali. Per quanto riguarda la nostra attività twitter, la lavorazione con questo tipo di algoritmo differisce leggermente rispetto ai vari passaggi elencati nei paragrafi precedenti. Uno dei problemi nell'utilizzare questo algoritmo è stato il fatto che pochissimi tweet sono effettivamente geolocalizzati. Di default, twitter non include la geolocalizzazione, e sono perciò gli utenti a decidere di condividere la loro posizione tramite le impostazioni del proprio profilo. Se avessimo lavorato sul file precedente, quindi in quello da 10.000 tweet, quelli localizzati sarebbero stati solo 800 circa, troppo pochi.
Di conseguenza ho preferito lavorare su un secondo file, questa volta da 290.000 nodi circa (tutti i tweet fra il 20 e il 22 maggio), per riuscire a selezionare 2.500 tweet localizzati. Nonostante siano la minoranza, forniscono materiale a sufficienza per mappare i dati.
Per lavorare con GeoLayout non servono relazioni fra i tweet, ma è sufficiente visualizzarli come punti su una mappa, di conseguenza il passaggio con R non è necessario e di può lavorare direttamente su file *.csv*. Ho eseguito un tentativo modificando lo script di R, salvando oltre a tweet e mentions anche le coordinate, ma ho ottenuto un file con una cinquantina di dati, decisamente poco utile in quanto comunque i tweet condivisi e geolocalizzati sono veramente troppo pochi rispetto alla mole di dati originale. Fondamentale da ricordarsi durante l'importazione del file è specificare che i campi

```
geo_coordinates_0, geo_coordinates_1
```

sono di tipo "double" e non di tipo "string" (ovvero numeri a 8 byte anziché parole), perché il layout prenderà in input solo quei due valori, e se non importati correttamente non riuscirà a mapparli.
Dopo il primo passaggio con GeoLayout, il grafo si presenterà come una minuscola Italia.

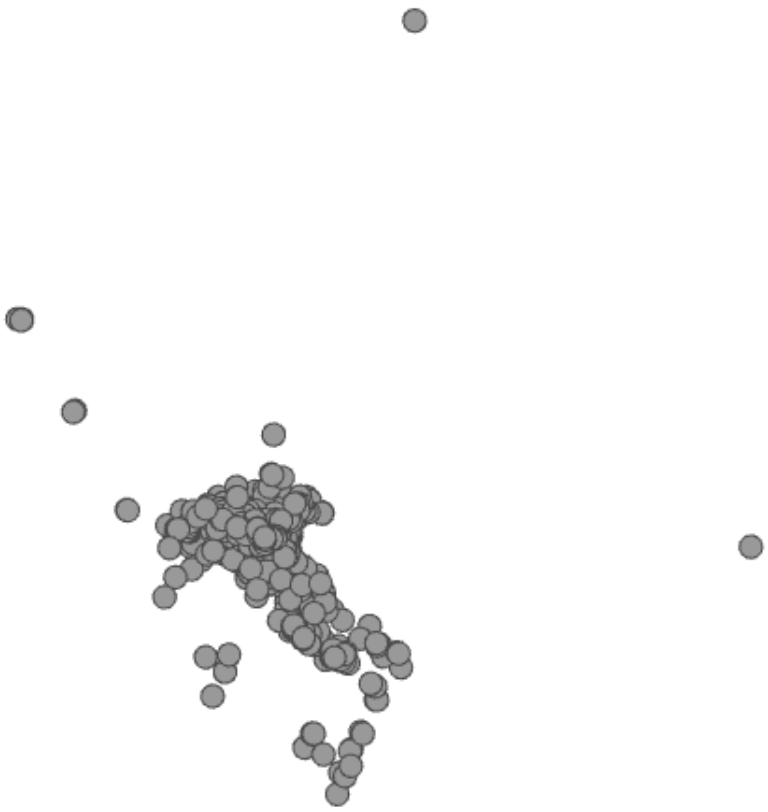


Figura 5.9: Una piccola Italia

Naturalmente serve maggiore dettaglio, di conseguenza è necessario modificare il parametro “Scale” da 1000 ad almeno 150.000. Aumentandolo, è come se si aumentasse il raggio virtuale della Terra, perciò numeri grandi corrispondono a tanti nodi sparsi con maggiore dettaglio rispetto a un raggio piccolo, dove i tweet appaiono tutti concentrati su un’area più piccola.

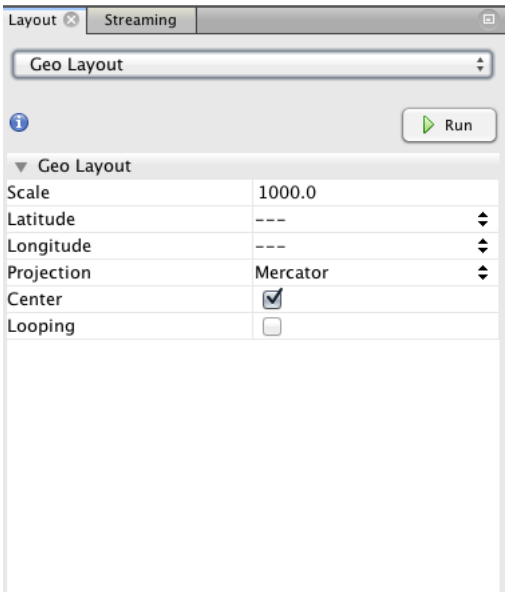


Figura 5.10: Il parametro Scale

Aumentando il raggio si inizia ad intravedere la concentrazione di tweet sull’Emilia Romagna, la cui forma si intuisce dal posizionamento stesso dei tweets. Si delineano le coste italiane, e piccole concentrazioni di tweet sopra le principali città italiane. Posizionando infine una mappa dell’Italia (con un programma esterno di grafica) in modo da avere i punti di riferimento, il risultato è come me l’aspettavo: la maggior concentrazione di tweet è giustamente intorno all’area dell’epicentro.

Figura 5.11: Con Scale a 150.000

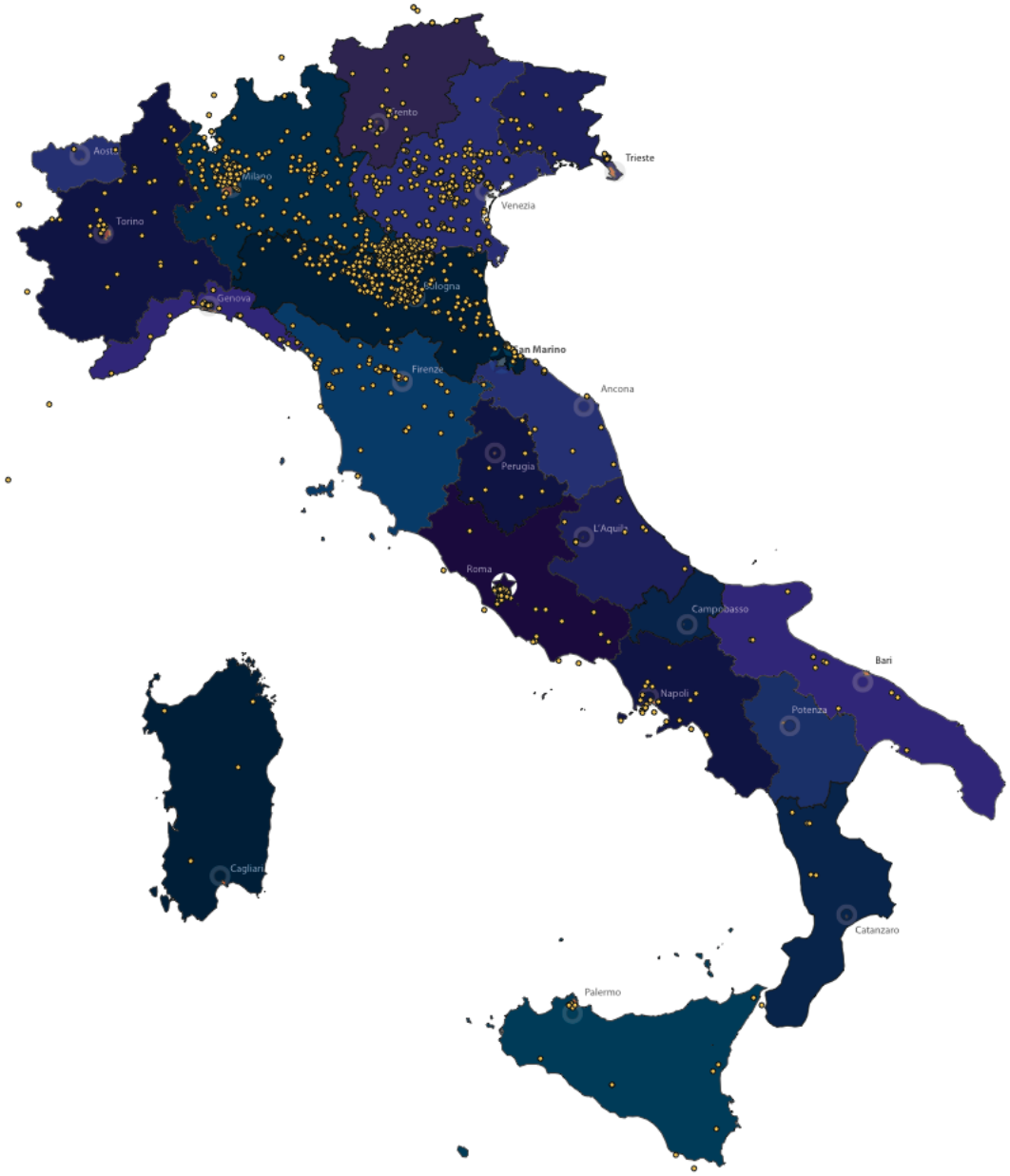


Figura 5.12: La mappa dei tweet italiani

5.7.2 La geolocalizzazione in Google Maps

Tramite Google Maps è possibile creare mappe personalizzate, ossia includere una lista di punti in un nuovo livello, in modo da poter visualizzare direttamente sulla cartina i propri punti di interesse. Per creare una nuova mappa personalizzata è possibile inserire i dati manualmente, posizionando dei pin semplicemente trascinandoli sulla zona che si vuole marcare, oppure importando una lista già completa di coordinate, che Google Maps processerà e farà apparire sulle mappe. Nel mio caso, avendo già una buona quantità di dati, ho preferito importare i dati tramite un file *.kml* , acronimo di Keyhole Markup Language, un file che deriva dall'XML. Per convertire quindi il file originale da *.csv* a *.kml* ho utilizzato un tool online, KMLTools, che ha generato il file nel formato richiesto da Google.
Caricando il file sulle mappe i punti sono apparsi in corrispondenza delle coordinate decimali. Il tool per la conversione del file tuttavia è molto sensibile, di conseguenza un piccolo errore all'interno delle coordinate porta all'eliminazione del dato, facendo apparire molti meno tweet rispetto alla mappa esportata direttamente da Gephi.

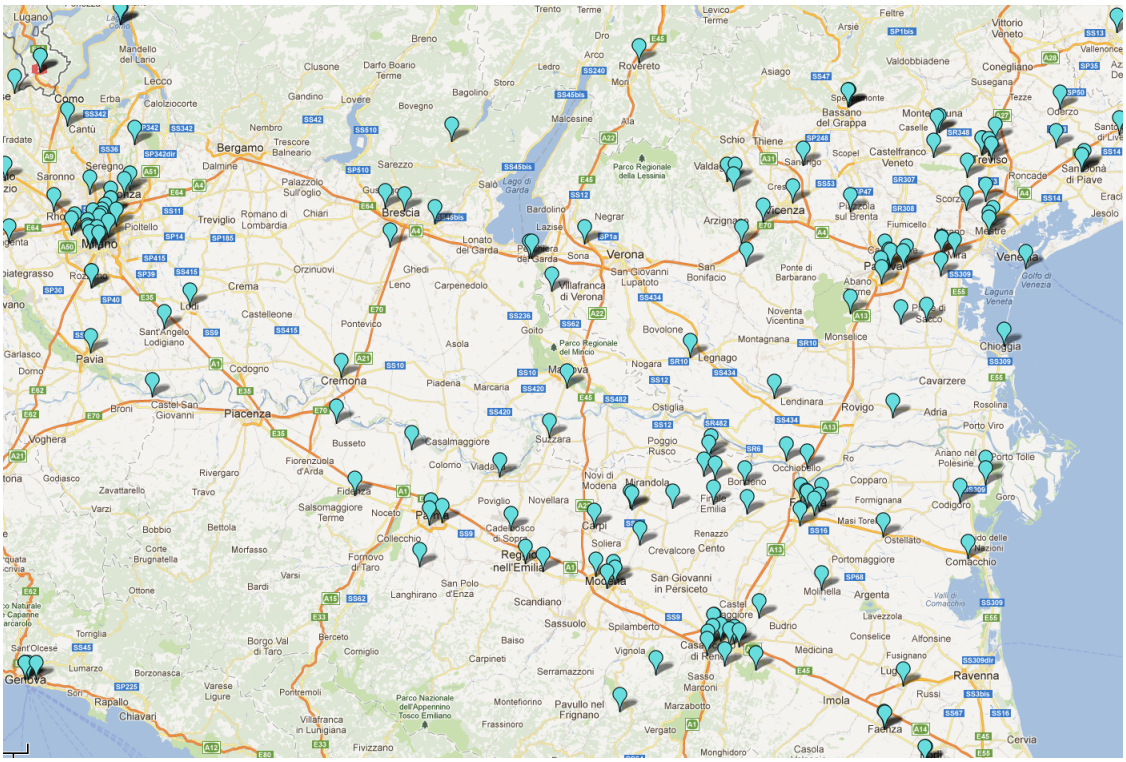


Figura 5.13: I tweet su Google Maps

5.8 Grafo dinamico

5.8.1 La teoria

Nonostante i grafi siano pensati tradizionalmente come oggetti statici, per quanto riguarda l'analisi dei social network o, più in generale, delle reti sociali, è noto che l'evoluzione nel tempo e il cambiamento del grafo dinamicamente possa riflettere meglio l'evoluzione delle interazioni sociali nei gruppi. Un modo per introdurre un cambiamento all'interno di un grafo è quello di assegnare al peso dei propri archi un valore di probabilità, quindi un arco con un valore di probabilità basso sarà rapidamente eliminato con l'avanzare del tempo. Uno dei problemi che si incontrano assegnando una probabilità agli archi è quello di ritrovarsi un grafo senza più connessioni o comunque senza connessioni a sufficienza per poter continuare ad elaborarlo. [13]
Nel nostro caso, la dinamicità del grafo è stata generata attraverso un metodo diverso, più adatto per studiare la diffusione di informazioni.

5.8.2 Dinamismo nel tempo

Il metodo utilizzato per studiare l'evoluzione del grafo di retweet è stato quello di associare ogni interazione degli utenti a un istante. Il file originale estratto da TwapperKeeper è stato processato attraverso uno script di *Gawk*, in modo che risultasse in uscita un altro file *.csv*, delimitato da virgole, del tipo:

```
user1, user2, timestamp1;timestamp2;timestamp3...
```

interpretabile come un tweet che parte da *user1* e arriva a *user2* nel momento *timestamp1*, oppure, se la comunicazione continua, nel momento *timestamp2*, *timestamp3* eccetera. Ogni *timestamp* è quindi un retweet dell'utente *user1* da parte dello *user2*. Se uno user ha condiviso più tweet di un altro, nella sua colonna *timestamp* ci saranno più valori. [14]
Lo script utilizzato per la conversione è il seguente:

```
1
2 # preparegexfattimeintervals.awk - Extract @replies for network
      visualisation
3 #
4 # this script takes a CSV archive of tweets, and reworks it into network
      data for visualisation
5 # data includes the timestamp of the tweet - multiple tweets between two
      users are concatenated with a semicolon
      (starttime1;starttime2;starttime3;...)
6 #
7 # expected data format:
8 # text,to_user_id,from_user,id,from_user_id,iso_language_code,source,
9 # profile_image_url,geo_type,geo_coordinates_0,geo_coordinates_1,
```

```
10 # created_at,time
11 #
12 # output format:
13 # source,target,starttimes
14 #
15 # script takes an optional offset argument, which is subtracted from the
     timestamp
16 # if offset is not set on the command line, the first tweet's timestamp
     is used instead
17 #
18 # Released under Creative Commons (BY, NC, SA) by Axel Bruns -
     a.bruns@qut.edu.au
19
20 BEGIN {
21
22 # populate the FILENAME variable
23     getline
24
25     print "source,target,starttimes"
26
27     ## first, build list of unique usernames in standard capitalisation
         style
28     ## (to avoid duplicates in different styles, e.g. @UserName vs.
         @username)
29     # skip header row
30     getline < FILENAME
31
32     i=1
33     # start first parse of input file
34     while(getline < FILENAME) {
35         nodes[i] = $3
36         # add any active tweeters
37         i++
38
39         if(starttime == 0) starttime = $13
40         endtime = $13
41
42         a=0
43         do {
44         # add all @reply recipients
45             match(substr($1, a),/@([A-Za-z0-9_]+)?/,atArray)
46             a=a+atArray[1, "start"]+atArray[1, "length"]
47             if (atArray[1] != 0) {
48                 nodes[i] = atArray[1]
49                 i++
50             }
51         } while(atArray[1, "start"] != 0)
52     }
53     close(FILENAME)
```

```
54
55      n = asort(nodes, sorted)
56      # sort and remove duplicates
57      j = 1
58
59      for (i=2; i<=n; i++) {
60          if(tolower(sorted[i]) == tolower(sorted[j])) {
61              delete sorted[i]
62          } else {
63              j=i
64          }
65      }
66
67      n = asort(sorted,sortednodes)
68      # sort remaining list
69
70  }
71
72  /@([A-Za-z0-9_]+)/ {
73  # main loop: match any line including @reply
74      if(!offset) offset = $13
75      # if not provided as script argument, set offset to first occurring
            tweet timecode
76
77      a=0
78      do {
79      # repeat for each @reply in the line (i.e. execute twice for '@user1
            @user2 tweet content')
80          match(substr($1, a),/@([A-Za-z0-9_]+)?/,atArray)
81          a=a+atArray[1, "start"]+atArray[1, "length"]
82
83          if (atArray[1] != 0) {
84              for(i in sortednodes) {
85              # match source and target users against sorted master list
                    of participating users in standard capitalisation
86                  if(tolower(sortednodes[i]) == tolower($3)) { source =
                        sortednodes[i] }
87                  if(tolower(sortednodes[i]) == tolower(atArray[1])) {
                        target = sortednodes[i] }
88              }
89
90              if(!edge[source "," target]) {
91                  edge[source "," target] = $13 - offset
92                  # create new edge[source,target] array entry, or
93              } else {
94                  edge[source "," target] = edge[source "," target] ";"
                        $13 - offset
95                   # add to existing entry
96              }
```

```
97          }
98
99      } while(atArray[1, "start"] != 0)
100
101 }
102
103 END {
104     for(i in edge) {
105         print i "," edge[i]
106     }
107 }
```

Sorge ora il problema del determinare la vita media di un tweet nel tempo, perché comunque il timestamp identifica solo il momento di nascita del retweet ma non la durata del collegamento. Per questo, ipotizzo in modo totalmente arbitrario un tempo di decadimento di un'ora per ogni tweet, cosicché ogni retweet stabilirà un collegamento fra due utenti che durerà un'ora. [14] Un ulteriore problema è che, nel file .csv, non è possibile esplicitare il momento di decadimento del tweet. Per questo è necessario ricorrere a un altro formato di file basato su XML, ossia .gexf. Permette infatti di definire intervalli temporali durante i quali esiste un collegamento. Per far questo convertirò il file .csv in un .gexf sempre attraverso Gawk, impostando come tempo di decadimento 3.600 secondi. Restituito il file .gexf potrò importarlo in Gephi. Questo script tiene anche in considerazione eventuali sovrapposizioni di retweet.

```
1
2 # gefxattimeintervals.awk - Process CSV of network edges and start times
      for network visualisation
3 #
4 # this script takes a CSV of network edges with attached start times
      (multiples allowed), and converts it into a GEXF for visualisation
5 #
6 # expected data format:
7 # source,target,starttime[;starttime2[;starttime3[;...]]]
8 #
9 # output format:
10 # GEXF network file. Weights of multiple edges whose timeframes overlap
      are increased.
11 #
12 # script takes an optional decaytime argument, which sets the expiry
      time of each edge - edge starting at time t expires at t + decaytime
13 # if decaytime is not set on the command line, decaytime is set to 100
14 #
15 # Released under Creative Commons (BY, NC, SA) by Axel Bruns -
      a.bruns@qut.edu.au
16
17 BEGIN {
18     getline
```

```
19      print "<gexf xmlns=\"http://www.gexf.net/1.1draft\"\n
            xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"\n
            xsi:schemaLocation=\"http://www.gexf.net/1.1draft\n

            http://www.gexf.net/1.1draft/gexf.xsd\"\n        version=\"1.1\">\n
            <graph mode=\"dynamic\" defaultedgetype=\"directed\">\n
            <attributes class=\"edge\" mode=\"dynamic\">\n     <attribute
            id=\"weight\" title=\"Weight\" type=\"float\"/>\n  </attributes>\n
            <edges>"
20
21      if(!decaytime) decaytime = 100
22  }
23
24  {
25      max = split($3,time,";")
26
27      for(i = 1; i <= max; i++) {
28          edgestart[i] = time[i]
29          edgeend[i] = time[i] + decaytime
30          edgeweight[i] = 1
31      }
32
33      if(max > 1) {
34          for(i = 2; i <= max; i++) {
35              top = max
36              for(j = 1; j <= max; j++) {
37                  if((edgestart[i] != edgeend[i]) && (edgestart[j] <=
                        edgestart[i]) && (edgeend[j] > edgestart[i]) && (j !=
                        i)) {
38                      edgestart[max+1] = edgestart[i]
39                      edgeend[max+1] = edgeend[i] < edgeend[j] ? edgeend[i]
                            : edgeend[j]
40                      edgeweight[max+1] = edgeweight[i] + edgeweight[j]
41
42  # two possibilities here: a) edge i starts inside the edge j timeframe,
        but finishes later; b) edge i timeframe is completely within edge j
        timeframe
43
44                      if(edgeend[j] > edgeend[i]) {
45                          edgeend[i] = edgeend[j]
46                          edgeweight[i] = edgeweight[j]
47                      }
48                      edgeend[j] = edgestart[max+1]
49                      edgestart[i] = edgeend[max+1]
50                      max++
51                  }
52              }
53          }
54      }
```

```
55
56      interval = weight = slice = ""
57      start = edgestart[1]
58      end = edgestart[1]
59
60      for(i = 1; i <= max; i++) {
61          if(edgestart[i] != edgeend[i]) {
62              if(edgestart[i] < start) start = edgestart[i]
63              if(edgeend[i] > end) end = edgeend[i]
64
65              slice = slice "              <slice start=\"" edgestart[i] "\"
                    end=\"" edgeend[i] "\" />\n"
66              weight = weight "                <attvalue for=\"weight\"
                    value=\"" edgeweight[i] "\" start=\"" edgestart[i] "\"
                    end=\"" edgeend[i] "\"/>\n"
67
68          }
69      }
70
71      print "      <edge source=\"" $1 "\" target=\"" $2 "\" start=\""
            start+0 "\" end=\"" end "\" weight=\"0\">"
72      print "          <attvalues>"
73      print weight
74      print "          </attvalues>"
75      print "          <slices>"
76      print slice
77      print "          </slices>"
78      print "      </edge>"
79  }
80
81  END {
82
83      print "    </edges>\n  </graph>\n</gexf>"
84
85  }
```

5.8.3 I Grafi dinamici in Gephi

Ora che il file è nel formato corretto, possiamo vedere come si comporta nel tempo. Importando il file in Gephi è fondamentale ricordarsi di mettere la spunta su "Create missing nodes", perché nel nostro file sono identificati solo gli archi, senza i nodi. Fortunatamente Gephi riesce a ricostruire facilmente i nodi a partire dagli archi.

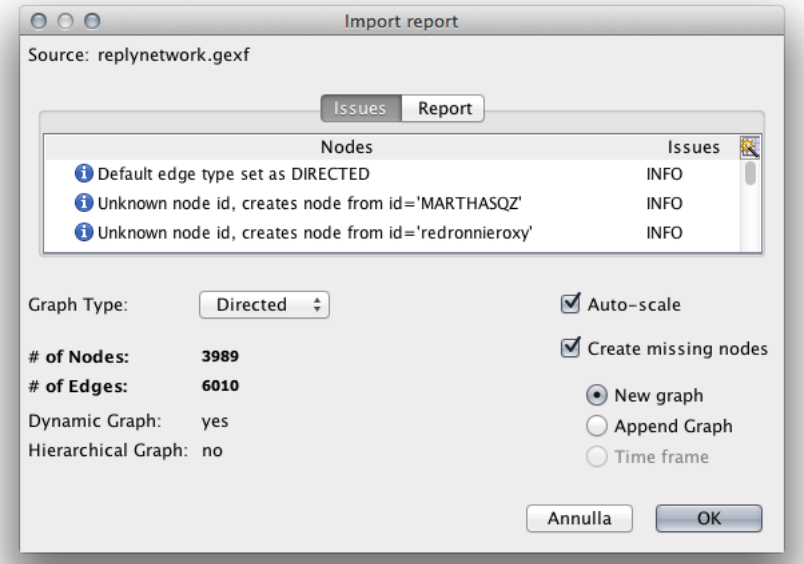


Figura 5.14: Create missing nodes

Nel caso in cui in network importato fosse molto grosso, il consiglio è quello di snellire il grafo applicando un filtro per eliminare dalla visualizzazione tutti i nodi di poca rilevanza, lasciando più spazio a quelli significativi. Tuttavia, in questo caso, ho preferito lasciare il grafo originale senza filtri, perché comunque manipolabile in modo piuttosto fluido.

Attivando la timeline, nella parte bassa della finestra di Gephi, il grafo mostrerà i suoi cambiamenti nel tempo o in un range di secondi che possiamo decidere a nostro piacimento. Per esempio dal millisecondo 0 al 2000, solo verso la parte finale, oppure dove si preferisce. Selezionando il range di valori la timeline cambierà di conseguenza, mostrando il range di tempo scelto evidenziandolo di verde.

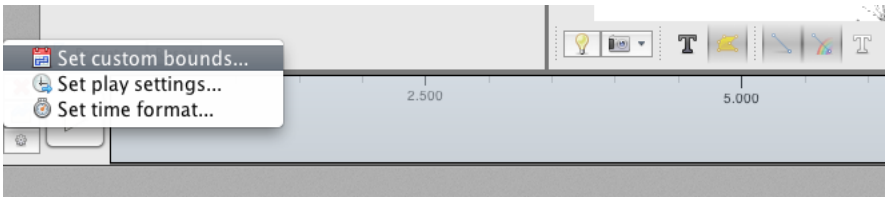


Figura 5.15: Menu per impostare i parametri della timeline

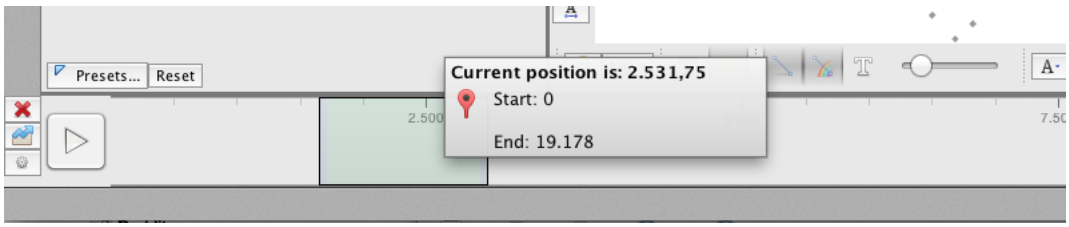


Figura 5.16: La timeline

Per un'esecuzione che mostri davvero il comportamento del grafo, è bene impostare prima di tutto un layout. In questo caso si è scelto *Force Atlas2* per la sua efficienza e per fare in modo che il grafo si muovesse più fluidamente possibile. Una volta lanciato il layout non basta che premere su play, e il grafo si modellerà con lo scorrere del tempo.
In particolare si nota come il grafo inizialmente si presenti come una rete casuale di nodi senza connessioni: gli utenti. Poco dopo la prima scossa qualche nodo inizia a connettersi con altri, e iniziano a formarsi i primi agglomerati grazie all'algoritmo lasciato in background. La situazione si evolve mostrando sempre più connessioni fra i nodi, fino a quando il traffico diventa veramente molto intenso e si formano grandi agglomerati e migliaia di archi. Il grafo è molto soggetto al cambiamento e si muove elasticamente per creare quello che sarà il risultato finale, molto simile infatti al grafo esportato in precedenza che mostrava la rete di retweet.
I parametri per visualizzare più lentamente o velocemente l'evoluzione del grafo si trovano nel menu in basso a destra, "Set play settings", dove è possibile decidere con quanto ritardo far muovere i nodi per poter visualizzare meglio le interazioni.

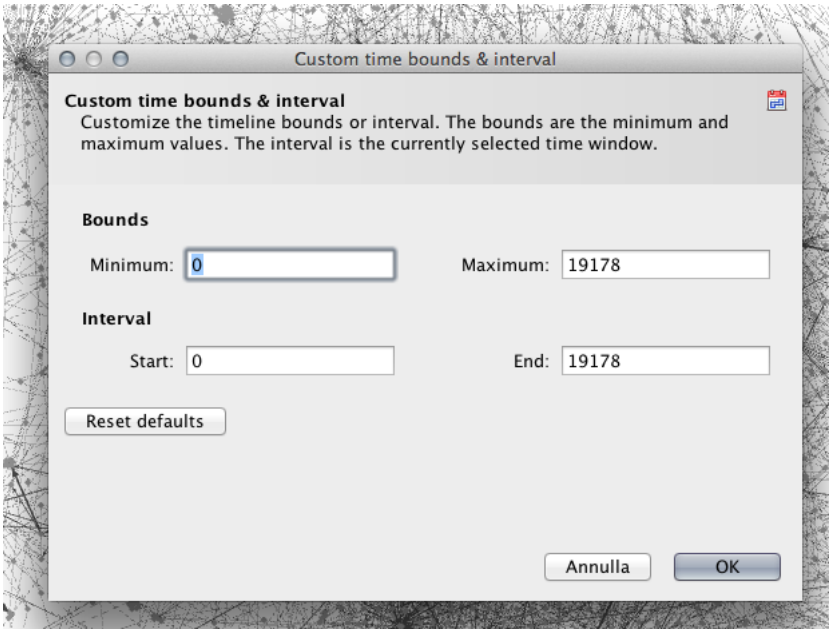


Figura 5.17: I bounds

Di seguito, qualche screenshot che mostra l'evoluzione del grafo.

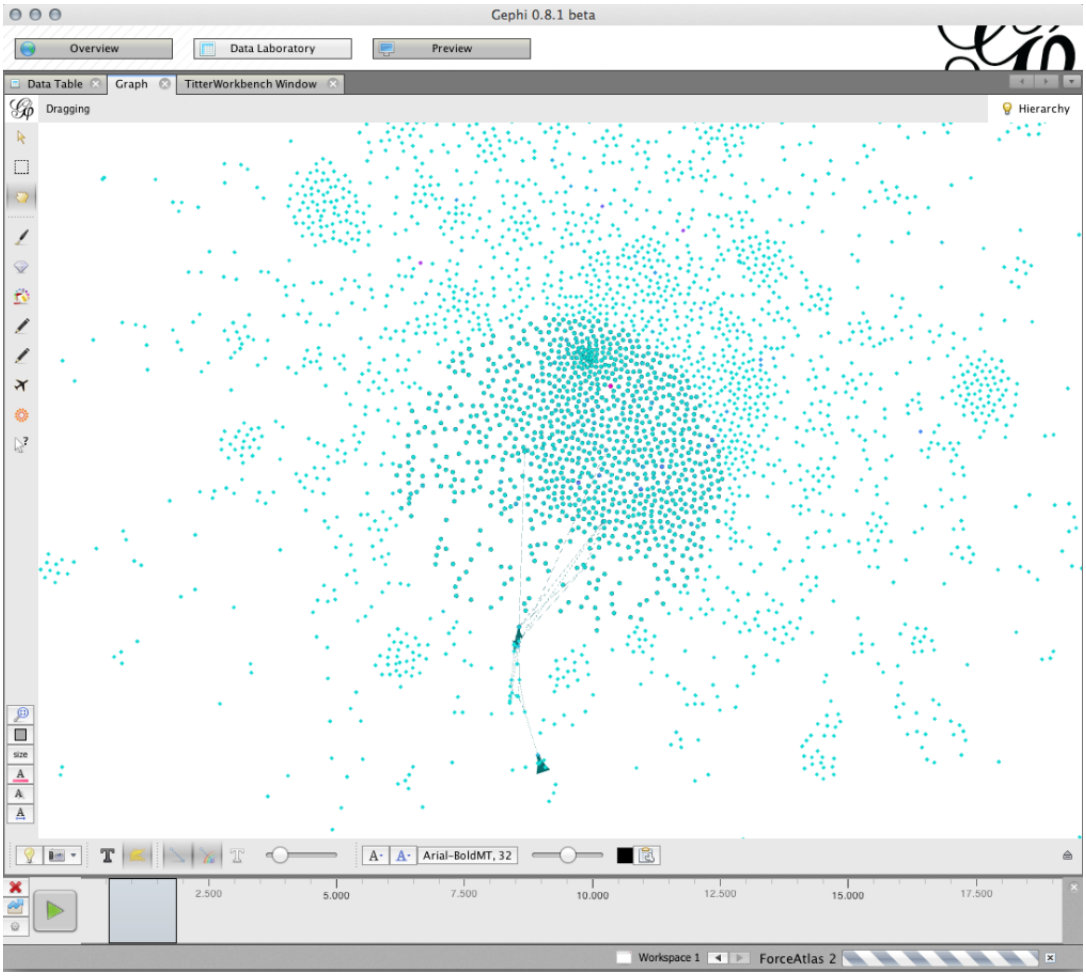


Figura 5.18: Prima di 2.500

5.8.4 Potenzialità

Senza alcun dubbio, porre in relazione un grafo rispetto al tempo può fornire una nuova dimensione di informazioni. Ipotizzando infatti una vita media per ogni arco il grafo mostrerà più precisamente la sua forma, che è quindi mutevole nel tempo, e riesce a fornirci un punto di vista più preciso e realistico di come gli utenti interagiscono fra loro. Un grafo statico infatti è in grado di mostrare le sue connessioni solo a istanti prefissati, e si presenta in una forma diversa a seconda dell'istante analizzato: potrebbe mostrarsi come un ammasso casuale di nodi se analizzato all'istante 0, o come una ragnatela di relazioni se analizzato all'istante 3. Molto spesso le analisi su un campione

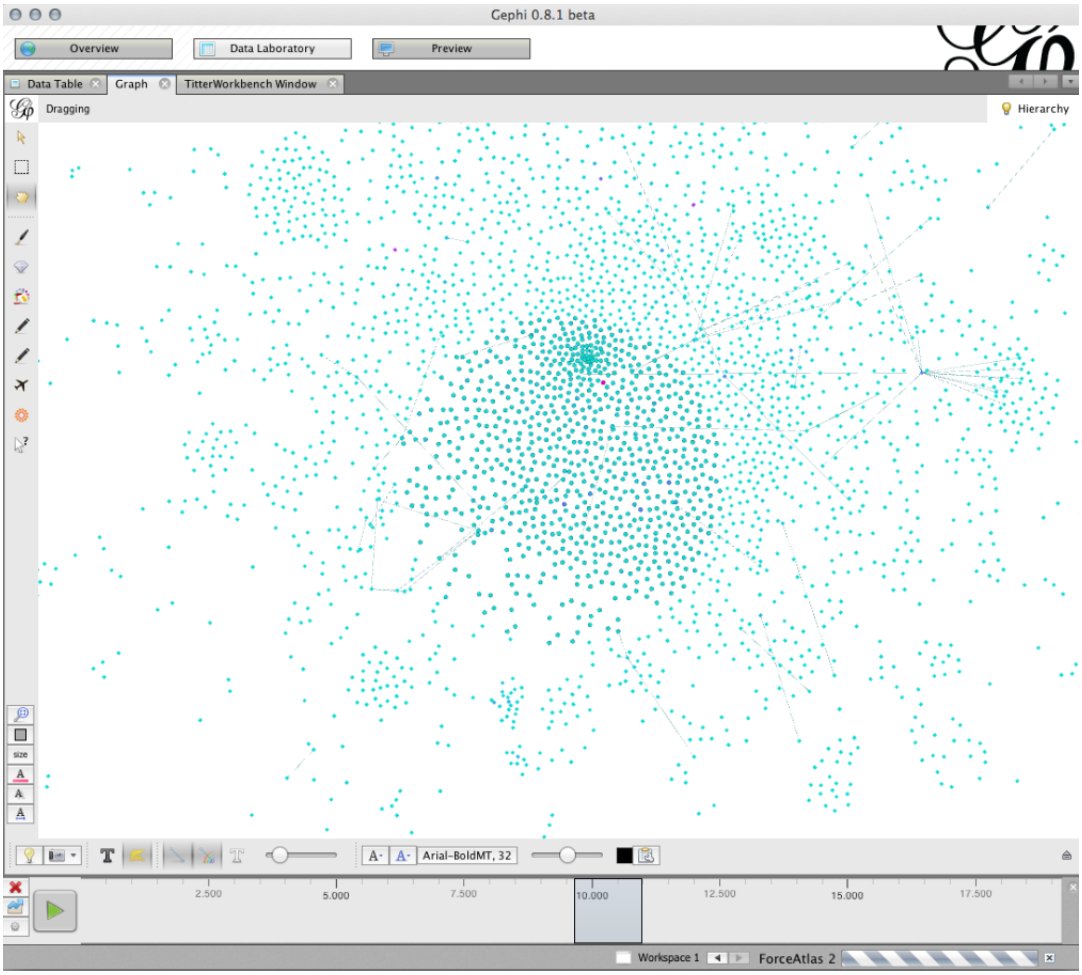


Figura 5.19: istante intorno a 10.000

di dati vengono realizzate all'istante temporale finale, realizzando quindi un grafo che rappresenta il momento finale del campionamento: in questo modo il grafo ottenuto conterrà di conseguenza fin troppe informazioni perché non riesce a tenere conto delle connessioni che si sono stabilite durante un piccolo arco di tempo e che sono poi decadute, ponendole allo stesso piano di altre connessioni più permanenti, sporcando il grafo con un surplus di informazioni. La parte interessante sta proprio nell'osservare come gli archi nascono, a partire da quale area del grafo, e di come l'informazione riesce a propagarsi come una reazione a catena, fornendo un quadro più completo di come la notizia è nata e cresciuta in 48 ore.

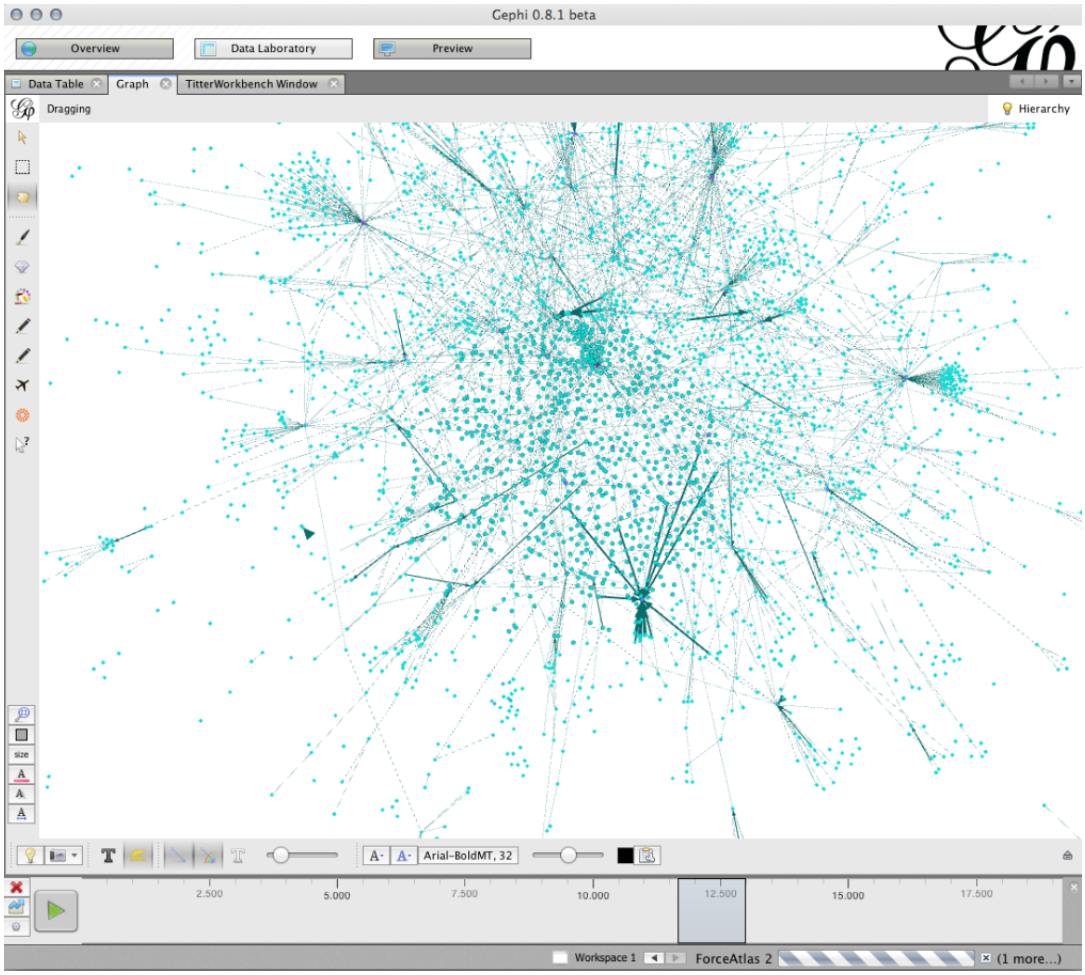


Figura 5.20: istante intorno a 15.000

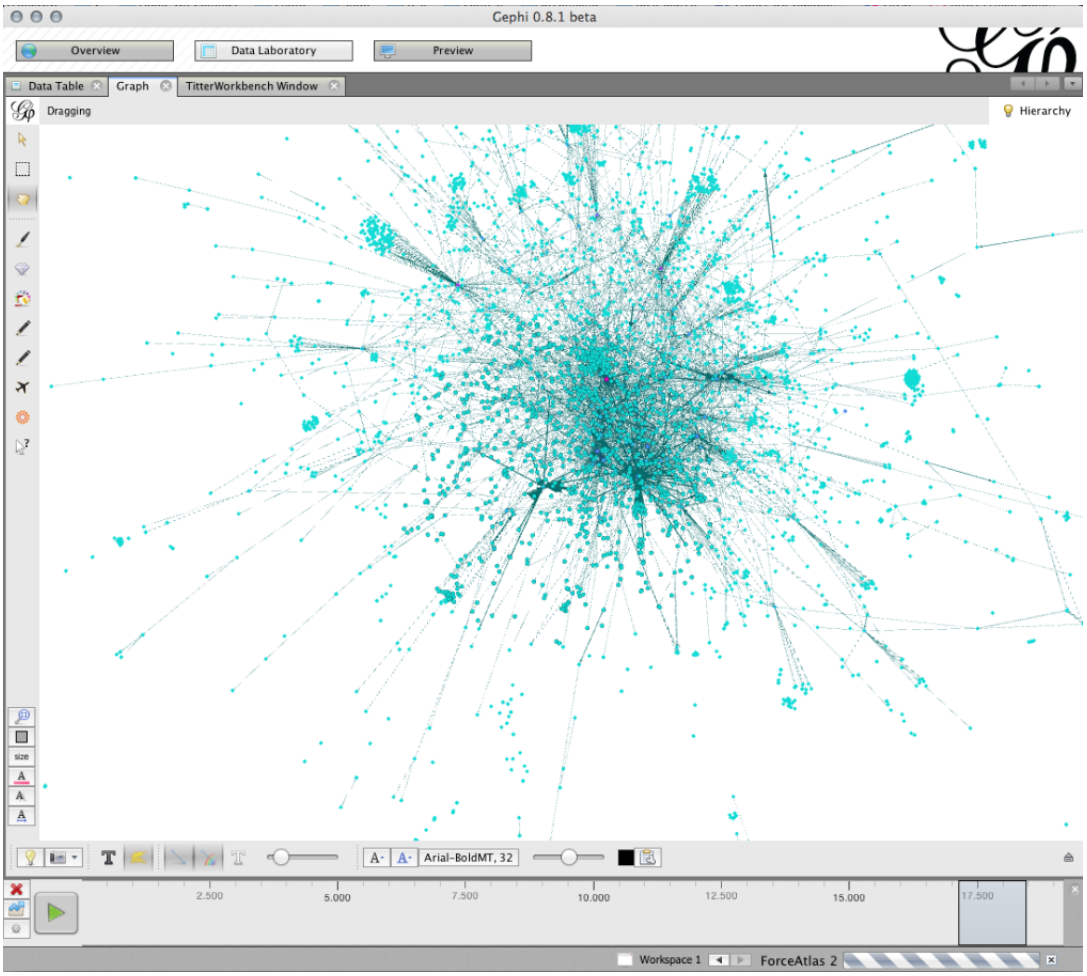


Figura 5.21: istante intorno a 17.500

Capitolo 6

Conclusioni

Gephi è stato molto utile per visualizzare il traffico twitter, semplificando in un grafo ciò che a prima vista non era altro che una collezione di dati. Assieme agli altri tool usati (R, Gawk, TwapperKepper) il risultato prodotto si è rivelato molto efficace dal punto di vista della comunicazione, e il prodotto di queste analisi può essere agevolmente studiato e diffuso tra gli operatori specializzati, quali disaster manager, esperti di marketing e via dicendo. Tuttavia però il procedimento per ricavare il grafo finale si è dimostrato essere piuttosto complesso e macchinoso, sia per colpa della molteplicità di linguaggi di programmazione utilizzati sia per l'insieme di formati file da convertire ad ogni passo. Sicuramente una semplificazione del processo aumenterebbe l'utilità del prodotto finale stesso, rendendolo accessibile a tutti e garantendo potenzialmente a chiunque la possibilità di analizzare piccole o grandi quantità di dati, sia per scopi di pura curiosità che di studio. La mia ricerca si muoverebbe quindi verso l'accessibilità di queste informazioni, migliorandone i processi di acquisizione e lavorazione dei dati.

La visualizzazione grafica permette di rendere le discipline testuali per antonomasia, come il giornalismo, complete ed efficaci. L'infografica, per esempio, non è solo una modalità accessoria per visualizzare i dati, ma uno strumento per comprendere al volo i fenomeni importanti, che regala anche la possibilità di studiarli e analizzarli in profondità. [15] L'esplosione dei dati e la loro circolazione ha portato a un naturale bisogno di tools per analizzarli, e la ricerca dovrebbe aiutare nel renderli più semplici e comprensibili. Le grafiche, in generale, aiutano moltissimo nella comprensione delle notizie e, in un mondo connesso come il nostro, i dati diventano un mezzo di espressione da non sottovalutare. In un mondo in continua evoluzione, anche l'informazione evolve e si velocizza sempre di più, parallelamente alla nostra capacità di assorbirla, analizzarla e rappresentarla.

Capitolo 7

Ringraziamenti

Vorrei evitare il paragrafo strappalacrime. Preferirei zero lacrime e persone commosse, in fondo è una laurea in informatica e l'unica che dovrebbe piangere sono io, per tutti questi anni persi davanti a un computer!

Ringrazio moltissimo tutti i miei professori, in particolare il prof. Danilo Montesi, in quale mi ha sempre coinvolto in progetti e attività stimolanti. Lo ringrazio per aver avuto fiducia in me e per aver sopportato l'enorme mole di e-mail a cui l'ho sottoposto, e anche per ogni ricevimento extra che ha fissato per me.

Un grande grazie a Luca Rossi, che mi ha dato l'accesso all'archivio dati TwapperKeeper di SNSItalia. Senza quei tweet la mia tesi non esisterebbe!

Ringrazio miei genitori. Naturalmente ogni figlio pensa sempre di avere i genitori migliori, beh: io li ho veramente. Quindi ringrazio mio babbo per avermi piazzato a 7 anni davanti a un computer Vectra e ad avermi insegnato le meraviglie della deframmentazione e di quanto è bello scoprire le cose per caso, per gioco. Se ho superato l'esame di statistica senza finire sommersa da un mare di formule è anche per un libro chiamato "Il Mago dei Numeri" che mi ha regalato da piccolissima in Autogrill, che mi ha fatto scoprire quanto è bella e intelligente la matematica. Ringrazio mia mamma per tutta la pazienza e le telefonate ansiose che mi arrivavano quando ancora facevo gli esami: è sempre stata talmente impaziente da chiamarmi non dopo gli esami, ma durante. La ringrazio anche per tutte le volte che mi ha dato modo di spiegarle come funziona un computer senza successo, per dimostrarmi che si vive benissimo anche senza, e che piuttosto il giardinaggio è un bellissimo hobby. Poi anche lei ha scoperto l'iPad, e ci ha un po' ripensato.

Ringrazio moltissimo Tommaso, che per quanto studi Giurisprudenza è sempre stato un informatico migliore di me, suscitando le mie ire notando che a lui le cose sono sempre

riuscite al primo colpo e a me mai. Sei stato fondamentale sotto (quasi) tutti gli aspetti: dal cucinarmi il pranzo tutti i giorni fino a sopportare i miei giorni di totale insofferenza. Quel quasi è per tutte le volte che mi hai svegliato mentre mi addormentavo studiando: sul momento non ero molto d'accordo. Sei sempre stato impeccabile e non hai mai sbagliato niente, e per questo motivo ti ringrazio e ti prego di continuare così, non potrei immaginare un compagno migliore.

I miei amici sono in assoluto la cosa più preziosa che ho. Carlotta, che mi accompagna dalla prima elementare, sempre pronta per intervenire ad ogni mio disastro e a consigliarmi come farebbe una sorella. Ringrazio tutte le nuove conoscenze fatte a Bologna, ormai fondamentali e immancabili in ogni mia giornata. Michele: senza il tuo incredibile potere di far ragionare le persone avrei abbandonato l'università dopo il primo anno. Martino: ti ringrazio per ogni messaggio ed e-mail che hai scritto velocemente e con mille errori di ortografia, restano fra le cose più divertenti che io abbia mai letto. Giacomo: l'unico ragazzo con un filo di lato artistico.. Grazie, almeno tu un po' mi capisci! Paolo: mi hai risollevato dai periodi più bui, perciò ti sarò sempre riconoscente, magari anche con un po' meno birra. Lorenzo: come enciclopedia vivente ti ringrazio per ogni dubbio che mi hai tolto. Sono tanti, te l'assicuro. Marco: per tutte le volte che abbiamo ordinato greco a domicilio e ci siamo sentiti in vacanza. Assieme a voi ringrazio naturalmente tutte le vostre donne, senza di loro non sareste sicuramente così incredibili: Carlotta C., Francesca e Ambra. Domiziana: l'unica conoscenza femminile a informatica, ti ho incontrata un po' per caso e ora è come se ti conoscessi da sempre. Ringrazio tutto il resto del DelirioErcolani per tutte quelle giornate passate sui libri assieme, e piano piano anche tutte le ore di svago, vacanza e amicizia: Nikolas, Simone, Ilaria, Marco M., Rita, Davide, Manuela, Marco V., Giulia, Iolao, Carla, Stefano e Federico, per avermi tenuto compagnia negli ultimi esami e avermi tenuto su di morale tutte le mattine.
Tutte le persone che con informatica non c'entrano nulla, ma che hanno migliorato le mie serate da quando vivo a Bologna: Marco C., Nico, Alice, Carlotta P., Clara.

Ringrazio Apple perché, da vera fangirl, ammiro i cambiamenti e le innovazioni che porta nella tecnologia.

Ringrazio, in due piccole righe qui sotto, anche la fotografia. Passione che non mi ha mai abbandonato e che continua ad essere viva nel mio percorso. Gianluca, tu sei incluso nella parola fotografia, per tutte le cose che mi hai insegnato e tutte le volte che mi hai fatto arrabbiare criticandomi, a ragione, una foto.

Spero di non aver dimenticato nessuno e concludo. Ah, forse una persona sì: ringrazio anche la me stessa che per l'ennesima volta mi ha dimostrato di essere forte e determinata, anche quando le cose si mettono male. Continua così!

Bibliografia

[1] Wikipedia.org, http://it.wikipedia.org/wiki/Grafo

[2] Wikipedia.org, http://it.wikipedia.org/wiki/Relazione_di_equivalenza

[3] Alan Bertossi, Alberto Montresor. Algoritmi e strutture dati, 2010.

[4] http://cvprlab.uniparthenope.it/staff/salvi/teaching_files/asd_files/2012/ASD-17%20(Visite%20di%20un%20grafo).pdf

[5] Gephi Tutorials. https://gephi.org/users/

[6] Patrick J. McSweeney. http://web.ecs.syr.edu/ pjmcswee/gephi.pdf, 2009.

[7] Evelien Otte, Ronald Rousseau. Social Network Analysis: a powerful strategy, also for the information sciences

[8] https://persuasionradio.wordpress.com/2010/05/06/using-netvizz-gephi-to-analyze-a-facebook-network/, 2010.

[9] Alan Mislove, Sune Lehmann, Yong-Yeol Ahn, Jukka-Pekka Onnela, J. Niels Rosenquist, http://www.ccs.neu.edu/home/amislove/twittermood/, 2010

[10] SNSItalia, http://snsitalia.wordpress.com/

[11] Wikipedia.org, http://it.wikipedia.org/wiki/Twitter

[12] Cornelius Puschmann. http://blog.ynada.com/339, 2010.

[13] D. D. Šiljak. http://www.ee.scu.edu/eefac/siljak/papers/Dynamic%20Graphs_Conf.pdf, 2006.

[14] Axel Bruns. http://mappingonlinepublics.net/2010/12/30/visualising-twitter-dynamics-in-gephi-part-2/, 2010.

[15] Geoff McGhee, http://www.misurarelacomunicazione.it/2011/03/11/limportanza-della-visualizzazione-grafica-journalism-in-the-age-of-data/, 2011.