
Università degli Studi di Salerno

Facoltà di Scienze Matematiche Fisiche e Naturali

---

Tesi di Laurea di Magistrale in

Informatica

# Identifying the sources of false information in social networks

**Relatore**
Prof. Vincenzo Auletta
Dott. Diodato Ferraioli

**Candidato**
Marco Amoruso
Matricola 0522500283

---

Anno Accademico 2015-2016

# Indice

# Capitolo 1

# Introduzione

## 1.1 Organizzazione della tesi

# Capitolo 2

# Concetti base

Per comprendere a fondo questo lavoro di tesi è necessario introdurre alcuni concetti, come le **social network** e la **teoria dei grafi**, i quali rappresentano l'ambito applicativo e la maniera di modellarlo.

## 2.1 Reti sociali

Una rete sociale è un costrutto teorico proveniente dalle scienze sociali, utilizzata per studiare le relazioni fra individui, gruppi, organizzazioni ed intere società. Il termine rete sociale viene utilizzato per descrivere una struttura sociale determinata dalle interazione tra gli attori che la compongono [1]. Tali reti sono quindi strutture relazionali tra attori ed in quanto tali costituiscono una forma sociale rilevante che definisce il contesto in cui si muovono quegli stessi attori. Una rete sociale risulta essere allora la struttura di relazioni, le cui caratteristiche potessero essere usate per spiegare, in tutto o in parte, il comportamento delle persone che la costituiscono. Elementi costitutivi della rete sociale sono dunque:

- I soggetti, che rappresentano le unità, i nodi che compongono la rete (individui, gruppi, posizioni, luoghi, etc.);
- Le relazioni che legano i soggetti, che compongono la rete e che possiedono determinate caratteristiche.

Con riferimento al contenuto della relazione è possibile cogliere ed individuare alcune particolari reti che, per il tipo di legami che le costituiscono, si caratterizzano per essere reti di sostegno (supporto sociale):

- *Formali*, costituite dalle istituzioni sociali;

- *Informali*, che non presentano una veste istituzionalmente definita;
- *Primarie*, costituite da quelle relazioni, che in virtù dei legami naturali, accomunano gli individui, come rapporti familiari, parentali, di vicinato;
- *Secondarie*, formate da relazioni di conoscenza indiretta;
- *Complesse*, composte da un elevato numero di relazioni con caratteristiche specifiche.

Nell'ambito delle scienze sociali, il concetto di rete sociale è stato utilizzato a lungo come "metafora" per tradurre quelle che sono le idee di società e d'azione sociale, come esito di vincoli ed opportunità emergenti dalle relazioni tra i soggetti. L'uso metaforico del termine ha posto il concetto di rete sociale ad un livello di astrazione piuttosto elevato, creando confusione terminologica e mancanza di chiarezza. Successivamente, mediante l'impiego scientifico del termine, tale livello di astrazione è diminuito, determinando il passaggio del concetto di rete dall'immagine intuitiva di un fenomeno complesso alla sua rappresentazione sul piano formale ed analitico. Ciò ha portato le reti sociali, come rappresentazione organizzativa dei rapporti sociali ed il suo metodo di studio, **l'analisi delle reti sociali**, ad essere adottate come strumenti teorici e metodologici per lo studio di numerosi fenomeni e processi. In ambito sociologico, tali studi hanno mostrato che nelle reti si depositano valori materiali, ma soprattutto non materiali che contribuiscono a determinare la "ricchezza" individuale e collettiva (diversa da individuo ad individuo, non solo grazie alle capacità relazionali, ma anche per effetto di specifici processi strutturali) [2]. Quindi, *l'analisi delle reti sociali* comprende principalmente tutta la parte teorica ed i modelli utilizzati per lo studio delle reti sociali in generale. Un esempio reale di una fitta rete sociale è quello mostrato in Figura 2.1. Di seguito, dopo alcuni cenni storici si passa alla descrizione di concetti e strumenti derivanti da una branca della matematica nota con il nome di *teoria dei grafi* che è alla base della *Social Network Analysis*.

### 2.1.1 Cenni storici

L'idea delle reti sociali nacque verso la fine del 1900, grazie alle teorie e le ricerche su gruppi sociali di *Èmile Durkheim* e *Ferdinand Tönnies*. Quest'ultimo sosteneva che i gruppi sociali possono esistere come legami personali e diretti, i quali collegano gli individui che condividono valori e credenze (*community*), oppure impersonali e formali (*società*) [3]. Durkheim fornì

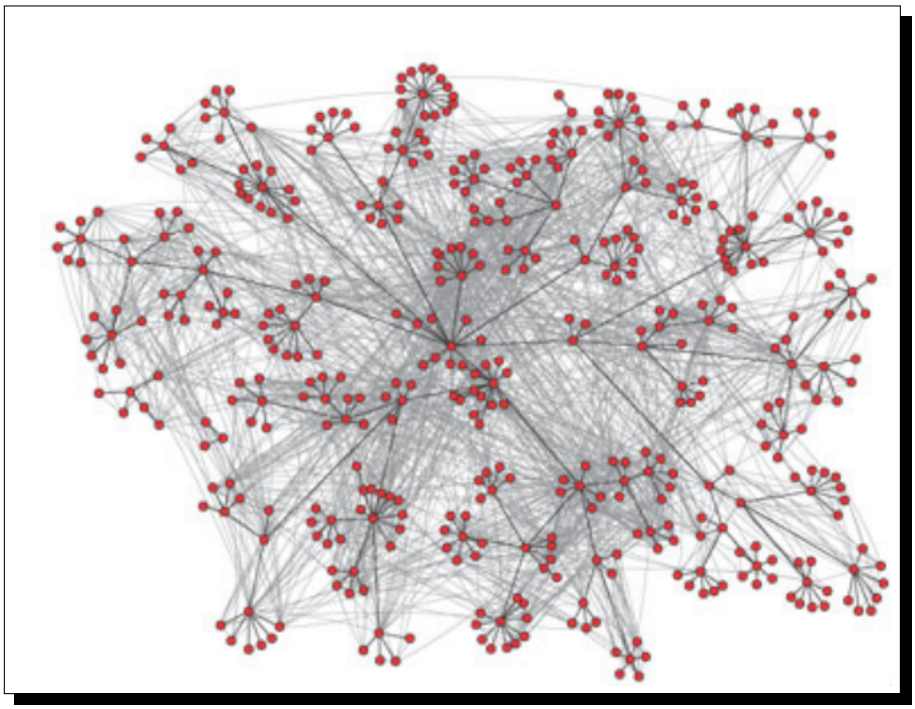

Figura 2.1: Scambio di e-mail tra i 436 impiegati di HP Research Lab

una spiegazione non individualistica dei fenomeni sociali, affermando che essi sorgono quando gli individui interagiscono costituendo una realtà che non può più essere rappresentata, in termini di proprietà dei singoli [4].
Inoltre intorno al ventesimo secolo, *Georg Simmel* si concentrò sulla natura delle reti e su quanto le dimensioni delle reti influenzassero le interazioni tra le componenti della rete, esaminandone le probabilità di interazione in reti debolmente collegate, piuttosto che in gruppi [5].
Importanti sviluppi nel campo possono essere riscontrati intorno al 1930, grazie a differenti ed indipendenti gruppi di psicologi, antropologi e matematici. In psicologia, *Jacob L. Moreno* iniziò una registrazione ed un'analisi sistematica delle interazioni sociali all'interno di piccoli gruppi. Per quanto riguarda l'antropologia, la fondazione della teoria delle reti sociali è dovuta ai lavori teorici ed entrografici di *Bronislaw Malinowski* [6], *Alfred Radcliffe-Brown* [7, 8], e *Claude Lévi-Strauss* [9], gruppo di antropologi sociali a cui vengono attribuiti i primi lavori di analisi di rete, investigando le community nel sud Africa, India e Regno Unito. In concomitanza a tale gruppo, l'antropologo inglese *S.F. Nadel* codificò una teoria riguardante le strutture

sociali che influenzò l'analisi delle reti [10]. Nel campo della sociologia, nei primi anni del 1930 *Talcott Parsons* preparò il terreno per l'adozione di un approccio relazionale per comprendere le struttura sociali [11, 12]. Successivamente, attingendo dalle teorie di Parsons, il lavoro del sociologo *Peter Blau* sulla teoria dello scambio sociale fornì un forte impatto per l'analisi dei legami relazionali [13–15]. A partire dal 1970, un numero crescente di studiosi lavorò per combinare le differenti tracce e tradizioni, fra cui un gruppo composto dal sociologo *Harrison White* ed i suoi studenti del dipartimento di scienze sociale dell'università di Harvard. Ed in maniera indipendente, *Charles Tilly* si concentrò sulle relazioni fra le dinamiche politiche e sociali, e *Stanley Milgram* sviluppò la tesi sui sei gradi di separazione [16]. In seguito, *Mark Granovetter* [17] e *Barry Wellman* [18] furono fra i primi studenti di White ad elaborare e sostenere l'analisi delle reti sociali [19–21].

### 2.1.2 Importanza ed applicazioni

Le reti sociali pervadono la nostra vita economica e sociale. Esse giocano un ruolo centrale nella trasmissione delle informazioni su opportunità di lavoro, nuovi prodotti, tecnologie, opinioni politiche e sono fondamentali per il commercio di beni e servizi. Le reti sociali sono anche importanti per determinare come le malattie si diffondono, quali prodotti compriamo, quali linguaggi utilizziamo per comunicare, come votiamo, così come la decisione di divenire dei criminali, quanta educazione abbiamo ricevuto o anche le nostre probabilità di successo professionale [22].
Di seguito alcuni campi d'applicazione delle reti sociali, per capirne la diffusione e l'importanza:

- **Scienze delle comunicazioni**, che affondano le proprie radici nelle scienze sociali ed umanistiche, descrivono i principi di communicazione alla base dello scambio di informazioni;
- **Criminologia** e **sociologia urbana**, in cui molta attenzione è stata posta sulla rete sociale formata da criminali. Ad esempio, *Andrew Papachristos* ha studiato gli omicidi fra bande come una serie di scambi di violenza fra di esse. Gli omicidi si diffondono solo da una singola banda, in quanto le altre più deboli non possono permettersi di uccidere i membri delle bande più forti attraverso rappresaglie, ma devono commettere altri atti violenti per mantenere la loro reputazione [23];
- **Sociologia economica**, la quale studia le cause e gli effetti sociali di fenomeni di natura economica dei singoli individui e/o gruppi. Sociologi, come Mark Granovetter, hanno sviluppato principi chiave sulle

interazioni fra strutture sociali, informazioni, abilità di punizione e ricompensa, e fiducia che ricorrono frequentemente in ambiti economici e politici [24];

- **Health care**, la quale utilizza le reti sociali non solo per quanto riguarda l'epidemiologia, ma anche per modellare l'educazione, la prevenzione contro le malattie, le diagnosi ed i trattamenti dei pazienti, così come i sistemi e l'organizzazione alla base dell'health care;

- **Linguistica**, la quale studia i linguaggi, la loro morfologia, lo scambio e le trasformazioni delle parole e dei suoni causati dall'interazione fra linguaggi e culture diverse;

- **Social media**, che rappresentano un nuovo modo di creare, condividere e scambiare informazioni, idee, contenuti multimediali in comunità virtuali. Attraverso tali strumenti, è possibile creare relazioni sociali fra individui che possono condividere interessi professionali e/o personali simili. Gli online social networks sono ampiamente diffusi e ne risulta importante l'analisi, in cui le informazioni si diffondono rapidamente, comprese le *false informazioni*, ed in cui numerosi fenomeni sociali possono essere osservati.

## 2.2 Teoria dei grafi

La teoria dei grafi è una branca della matematica, che permette di descrivere insiemi di oggetti e le loro relazioni. Un **grafo** è il mezzo attraverso il quale è possibile specificare relazioni tra una collezione di oggetti. Esso consiste di oggetti chiamati *nodi* o *vertici* e di relazioni tra alcune coppie di nodi, chiamate *archi* (una rappresentazione di grafo in 2.2).

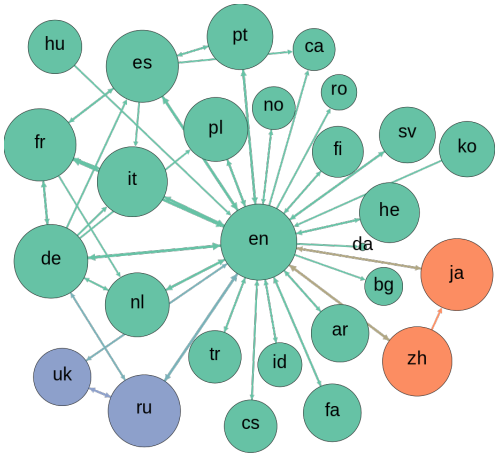


Figura 2.2: Wikipedia Multilingual Network Graph (July 2013)

L'origine storica della teoria dei grafi è generalmente fatta risalire ad un lavoro sviluppato da *Eulero* nel 1736, in cui veniva data una risposta ad un famoso quesito matematico noto come problema dei ponti di Königsberg. Königsberg era una ricca e popolosa città della Prussia, che si sviluppò sulle rive del fiume Pregel e sulle due isole formate dal fiume in quell'area, che nel 1700 era attraversata da sette ponti (come mostrato in 2.3).
Il quesito a cui rispose Eulero era se fosse possibile progettare un percorso che permettesse a ciascun abitante di partire da casa e tornarvi, dopo aver attraversato ciascun ponte una ed una sola volta. Per dimostrare che il problema non aveva soluzioni sostituì ogni riva del fiume e ogni isola con un nodo ed ogni ponte con un arco; in tal modo trasformò il processo di ricerca di una soluzione del problema nell'analisi e nello studio topologico del grafo così costruito.

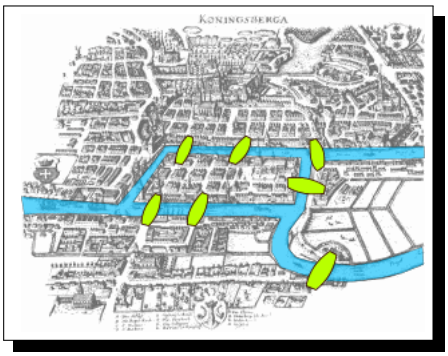


Figura 2.3: Rappresentazione della città di Königsberg nel 1700

La relazione tra una coppia di nodi può essere di due tipi:

- *Simmetrica*: l'arco connette i nodi con un collegamento bidirezionale ed è detto *indiretto*. Un grafo costituito di soli archi indiretti è anch'esso detto indiretto.
- *Asimmetrica*: l'arco connette i nodi con un collegamento unidirezionale ed è detto *diretto*. Un grafo costituito di soli archi diretti è anch'esso detto diretto.

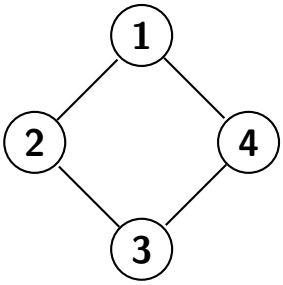


Figura 2.4: Grafo indiretto

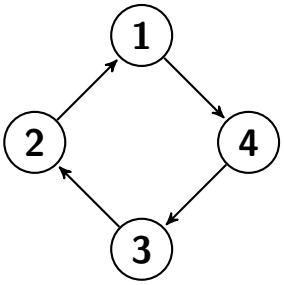


Figura 2.5: Grafo diretto

### 2.2.1 Definizione formale

Un grafo può essere formalmente descritto come una coppia di insiemi **G = (V, E)**, dove V è l'insieme dei nodi ed E è l'insieme degli archi. Un arco e ∈ E è rappresentato come un sottoinsieme di due elementi di V, $e = \{u, v\}$ per $u, v \in V$. Dato l'arco $e = (u, v)$, i nodi $u$ e $v$ sono detti *estremi* di $e$, e si dice che l'arco $e$ *incide* su $u$ e $v$ e che $u$ e $v$ sono *vicini* o *adiacenti*. Si definisce *intorno* o *vicinato* di un nodo $v$ in $G$, indicato con $N(v)$, l'insieme dei nodi adiacenti a $v$.

Le rappresentazioni atte a descrivere un grafo sono molteplici:

- *Rappresentazione grafica*: ad ogni nodo corrisponde una figura circolare sul piano e ad ogni arco (i, j) corrisponde una linea che collega il nodo i al nodo j.
- *Matrice di adiacenza*: matrice di dimensione $n \times n$, dove $n$ è il numero di nodi, il cui elemento (i, j) assume valore 1 se esiste l'arco tra il nodo i ed il nodo j, 0 altrimenti.
- *Lista di adiacenza*: ad ogni vertice $v$ è associata la lista dei nodi ad esso vicini.

### 2.2.2 Concetti utili

Negli anni, gli studi sulla teoria dei grafi hanno prodotto una quantità enorme di definizioni e teoremi, per cui, di seguito vengono descritti solamente i concetti necessari alla comprensione di questo lavoro di tesi.

**Multigrafo.** Un multigrafo $G = (V, E)$ è un grafo a cui è permesso avere archi multipli, ovvero due o più archi che sono incidenti sugli stessi due vertici.

**Grafo connesso.** Un grafo è connesso se, per ogni coppia distinta di vertici $(u, v)$, esiste un cammino da $u$ a $v$.

**Grafo pesato.** Un grafo $G$ si dice pesato se ad ogni arco $e$ è associato un numero reale $w(e)$, chiamato il suo *peso*.

**Sottografo.** Un grafo H si dice sottografo di un grafo G se i vertici di H sono un sottoinsieme dei vertici di G e gli archi di H sono un sottoinsieme degli archi di G. Siano $G = (V, E)$ ed $H = (V_1, E_1)$ due grafi. H è un sottografo di G se e solo se $V_1 \subseteq V$ ed $E_1 \subseteq E$.

Si definisce *sottografo indotto* da $V_1 \subseteq V$ in $G$ il grafo $H = (V_1, E_1)$, dove l'insieme degli archi $E_1$ è tale che un qualsiasi arco $(u, v)$ appartiene all'insieme se i nodi $u$ e $v$ appartengono a $V$ ed $(u, v) \in E$.

**Grado di un nodo.** Il grado di un nodo $v$ è il numero di nodi ad esso adiacenti ed è indicato con *deg(v)*.
In un grafo diretto, si distinguono due tipi di grado:

- *in-deg(v)*, il grado in ingresso del nodo $v$, dato dal numero di archi in cui $v$ compare come nodo destinazione;
- *out-deg(v)*, il grado in uscita del nodo $v$, dato dal numero di archi in cui $v$ compare come nodo sorgente.

**Cammino.** Un cammino è una sequenza di nodi, in cui ogni coppia consecutiva della sequenza sia connessa da un arco. Formalmente, un cammino è una sequenza di vertici $v_0, v_1, \ldots, v_n \in V$ tale che $\{v_{i-1}, v_i\} \in E, \forall 1 \leq i \leq n$. Un cammino con almeno tre vertici distinti, i cui vertici di inizio e fine coincidono, è detto *ciclo*. Quindi un ciclo è una sequenza di nodi $v_0, v_1, \ldots, v_n \in V$ tale che $v_0 = v_n$.

**Componente connessa.** Una componente connessa di un grafo G è un sottoinsieme di nodi del grafo, in cui per ogni vertice del sottoinsieme esiste un cammino che lo collega ad ogni altro nodo.

**Albero.** Un albero è un grafo connesso che non contiene cicli come sottografi, quindi esiste un unico cammino da un nodo ad un altro.

**Foresta.** Una foresta è un grafo in cui ogni componente connessa è un albero.

**Spanning tree.** Uno spanning tree o albero ricoprente è un sottografo $H = (V_1, E_1)$ di un grafo $G = (V, E)$ che è un albero, dove $V_1 = V$. Uno spanning tree minimo è uno spanning tree di un grafo pesato, nel quale sommando i pesi degli archi si ottiene un valore minimo.

**Arborescence.** Un'arborescence è un sottografo diretto $G = (V, E)$, il quale contiene esattamente un unico cammino diretto da un nodo specifico $u$ ad ogni altro vertice $v \in V'$, dove $V' \subseteq V$. Il nodo $u$ viene chiamato *radice*, in quanto non ha nessun arco entrante. Una arborescence è una *spanning arborescence* se l'insieme dei vertici $V'$ è pari a $V$.

**Branching.** Una branching $B$ di $G = (V, E)$, dove $G$ è un grafo diretto, è una foresta di arborescence disgiunte.

# Capitolo 3

# Diffusione di false informazioni

Quando le persone sono connesse da una rete, diventa possibile per loro influenzare le decisioni ed i comportamenti degli individui facenti parte della rete. In una vasta gamma di situazioni le persone sono influenzate da altre: le opinioni che hanno, i prodotti che comprano, le posizioni politiche che sostengono, le attività che perseguono, e molto altro; in tali occasioni sorgono diversi processi sociali, in cui le reti costituiscono il tramite per l'aggregazione del comportamento dei singoli in risultati collettivi comuni.
Come primo esempio, supponendo di dover scegliere un ristorante in una città non conosciuta, e basandosi sulle proprie ricerche si intende andare al ristorante $A$. Tuttavia, arrivando al ristorante si nota che nessuno sta mangiando, a differenza del ristorante vicino $B$ che è quasi pieno. Se si crede che le altre persone abbiano gusti simili ai propri e che abbiano informazioni migliori su dove mangiare, diventa una scelta razionale seguire la folla al ristorante $B$, piuttosto che seguire le proprie informazioni. Per cui l'informazione che si può inferire dalle persone al ristorante $B$ può essere più potente della propria. In tal caso, si dice che un fenomeno di *information cascade* si è verificato. Generalmente, un information cascade si verifica quando le persone compiono decisioni sequenzialmente. In tale scenario le persone osservano le azioni dei predecessori e da queste inferiscono informazioni.
L'imitazione dei comportamenti degli altri non avviene solamente a causa di informazioni limitate, ma può avvenire ad esempio in circostanze di pressione sociale e conformismo. Esistono diverse motivazioni razionali per cui si imitano i comportamenti degli altri, generalmente la principale è dovuta al fatto che emulare le persone porta un beneficio maggiore rispetto al costo

che comporta tale azione. Ad esempio, ritornando all'esempio dei ristoranti, se si sceglie di andare al ristorante popolare $B$, questo comporta che il beneficio ottenuto supera il costo da pagare, ovvero la lunga attesa [25].
Quando si modellano i processi dai quali nuove idee ed innovazioni vengono adottate, la rete sociale sottostante può essere considerata concettualmente in due livelli:

- Una popolazione relativamente amorfa di individui, dove si è interessati agli effetti nel complesso;
- Una rete considerata dal punto di vista strutturale, in cui si analizza come i nodi individuali siano influenzati dai loro vicini.

In questo lavoro di tesi si è interessati al secondo punto di vista, quindi al processo decisionale dei singoli individui, da cui un determinato comportamento può iniziare a diffondersi attraverso i collegamenti della rete, tale fenomeno prende il nome di *cascading behaviour*. In questo scenario gli individui devono scegliere fra diverse azioni da intraprendere e sono influenzati in maniera proporzionale dalle scelte dei vicini.
Molte delle nostre interazioni con il resto del mondo avvengono in maniera locale, piuttosto che globale. Infatti siamo più preoccupati a riguardo delle decisioni degli amici, dei colleghi e delle persone vicine, rispetto alle decisioni di tutta la popolazione. Ad esempio, in un ambiente di lavoro scegliamo tecnologie compatibili con le persone con cui collaboriamo, anziché utilizzare le tecnologie più popolari globalmente. Similmente, adottiamo punti di vista politici che sono allineati con quelli dei nostri amici, anche se possono costituire la minoranza. Tale fenomeno è stato studiato inizialmente da sociologi ed intorno al ventesimo secolo stabilirono le strategie di base per analizzare la diffusione dei comportamenti ed i fattori che ne facilitano od impediscono il progresso. Come ad esempio *Ryan* e *Gross*, i quali studiarono l'adozione di un seme di mais ibrido fra i contadini negli Stati Uniti, mediante interviste per capirne il come ed il quando. Scoprirono che molti agricoltori vennero a conoscenza del seme ibrido dai venditori e che molti di loro si convinsero a provarlo sulla base delle esperienze dei vicini nella loro comunità.
Inoltre tale processo è importante anche perché in grado di descrivere la diffusione di malattie epidemiche, le quali si trasmettono da persona a persona.

Come già detto in precedenza, gli individui formano le loro credenze sulla base delle informazioni che ricevono da altre persone, come amici, vicini, colleghi. Un aspetto cruciale da affrontare è se il processo di scambio di in-

formazioni possa portare alla formazione di credenze più accurate oppure a determinati errori e confusioni o alla diffusione di **false informazioni**. Con l'avvento dei social media, lo scambio di informazioni ha avuto un incremento esponenziale, sia in termini di grandi quantità di dati che nella velocità di diffusione [26]. False informazioni possono essere diffuse non intenzionalmente, in tal caso si parla di *misinformation*, oppure in maniera intenzionale, trattandosi di *disinformation*. La diffusione di false informazioni segue gli stessi meccanismi descritti precedentemente di cascading behaviour ed è importante modellarla ed analizzarne le dinamiche, in quanto ha un forte impatto sia dal punto di vista sociale che economico.

## 3.1 Cause ed impatti

Il *World Economic Forum* ha elencato la *"massive digital misinformation"*, come uno dei maggiori rischi per la società moderna [27]. Le conoscenze, opinioni, credenze e percezioni delle persone sul mondo e sulla sua evoluzione si formano e si modellano, attraverso le informazioni a cui possono accedere, la maggior parte delle quali provenienti da giornali, televisione [28], e, più recentemente Internet. Internet rimane un inesplorato territorio in rapida evoluzione. Le generazioni attuali sono in grado di comunicare e condividere informazioni istantaneamente ad una scala più ampia rispetto al passato, soprattutto mediante i social media. Mentre i benefici sono evidenti e ben documentati, il nostro mondo iperconnesso potrebbe anche consentire la diffusione rapida e virale di informazioni che sono intenzionalmente o non intenzionalmente fuorvianti e provocatorie, con gravi conseguenze.
Internet non ha dei meccanismi di autocorrezione, come dimostra *Wikipedia*, in cui chiunque può disseminare false informazioni, anche se una comunità di volontari di Wikipedia di solito trova e corregge gli errori rapidamente. L'esistenza di breve durata di false informazioni sul sito difficilmente provoca gravi conseguenze nel mondo reale; tuttavia, è concepibile che una falsa informazione si possa diffondere viralmente ed avere un impatto devastante, prima di essere efficacemente corretta. È altrettanto probabile che l'autore originale del contenuto offensivo non sia a conoscenza del suo uso improprio oppure che sia stato innescato da un errore di traduzione. L'avvento dei social media ha reso ogni utente un auto-editore senza revisione, controlli di accuratezza dei fatti ed alcuna responsabilità. I fatti che vengono presentati risultano veri e certificati per milioni di utenti, soltanto perché vederli sui propri schermi rappresenta una garanzia di veridicità. La diffusione di false informazioni si basa generalmente sull'ignoranza e sulla credulità delle

persone. Molto dipende dalla falsa informazione stessa, dal suo significato, dalla rilevanza, dal contesto ed ovviamente dalla capacità di non essere etichettata come tale.
Ecco alcuni esempi di false informazioni con i relativi impatti:

- Un musicista che aveva viaggiato con *United Airlines*, fece un reclamo per i danni subiti alla sua chitarra e quando la società affermò che era già rotta, scrisse e cantò la canzone "United Breaks Guitars" e la caricò su YouTube. Il video divenne virale e la compagnia perse il 10% del valore azionario [29, 30];

- Nel novembre 2012, la BBC trasmise una denuncia su abusi sui minori in cui un politico era apparentemente coinvolto. L'identità del politico fu facilmente scoperta su Twitter, il quale fu oggetto di circa 10.000 tweet. In cima alle azioni legali contro tutte le persone che avevano diffuso queste false informazioni su Twitter, il politico diffamato ricevette 185.000 sterline per danni da parte della BBC [31, 32];

- L'esistenza su YouTube di un video intitolato "Innocence of Muslims", caricato da un cittadino negli Stati Uniti, ha scatenato rivolte in tutto il Medio Oriente, causando più di 50 morti [33];

- Un utente Twitter impersonificando il ministro dell'interno russo Vladimir Kolokoltsev nel luglio 2012 diffuse la falsa notizia che il presidente siriano Bashar al-Assad fosse stato ucciso o ferito, causando l'incremento dei prezzi del greggio [34];

- Trentamila persone originari dell'Assam fuggirono dal centro tecnologico di Bangalore in preda al panico nel 2012, dopo la ricezione di messaggi che sostenevano che sarebbero stati attaccati in risposta a violenze provenute da parte del loro paese d'origine [35, 36].

## 3.2 Modelli di diffusione

Poiché questo lavoro di tesi ha l'obiettivo di identificare le sorgenti di false informazioni è necessario trovare una strategia che ne simuli il processo di diffusione. In letteratura, sono noti diversi modelli di diffusione dell'informazione, i quali hanno origine da reti sociali, sistemi di particelle che interagiscono, epidemiologia, etc.
In questa sezione vengono descritti alcuni dei più importanti, in particolare i primi tre modelli derivano dall'epidemiologia.

### SI

Nella rappresentazione matematica le dinamiche di un'epidemia si possono ridurre alla transizione tra uno stato e l'altro della malattia. In questo modello, che è il più semplice tra quelli appartenenti a tale classe, sono presenti due stati: *susceptible* e *infected*. Un individuo che si trova nello stato susceptible è colui che non ha la malattia ancora, ma potrebbe contrarla se entrasse in contatto con qualcheduno affetto da essa. Mentre un individuo nello stato infected è qualcuno che ha la malattia e che potenzialmente può passarla a persone nello stato susceptible [37]. Considerando la diffusione di una malattia in una popolazione di $n$ individui, sia $S$ il numero medio di individui nello stato susceptible e $X$ il numero medio di infetti, l'interazione casuale fra gli individui in $S$ e quelli in $X$ viene regolata in base ad un fattore $\beta$, il quale indica il numero medio di contatti fra un individuo susceptible con uno infected. Per cui, un individuo nello stato infected ha un contatto in media con $\beta S/n$ persone nello stato susceptible. Quindi siccome il numero medio di infected è $X$, il tasso globale di nuove infezioni sarà $\beta XS/n$.

### SIR

Nel modello SI gli individui una volta infetti lo saranno sempre. Tuttavia per molte malattie, le persone possono guarire dopo certo periodo di tempo, in quanto i loro sistemi immunitari combattono contro gli agenti che causano la malattia. Inoltre, spesso le persone conservano la loro immunità dopo esser guariti dalla malattia, così da non poterla contrarre nuovamente. Per rappresentare questo processo viene utilizzato un terzo stato, denominato *recovered*. La dinamica di questo modello prevede due fasi. Nella prima, gli individui nello stato susceptible passano nello stato infected, quando entrano a contatto con individui infetti. Mentre nella seconda fase, gli individui infetti possono guarire dalla malattia (o morire) in base ad una costante $\gamma$.

### SIS

Una differente estensione del modello SI è quella che permette ad un individuo di poter contrarre nuovamente la malattia. Tale aspetto modella il processo in cui le vittime che riescono a riprendersi dalla malattia, non ottengono immunità da quest'ultima, per cui possono essere reinfettati. Il modello SIS si divide anch'esso, come il SIR, in due fasi. Nella prima gi individui nello stato susceptible, che entrano in contatto con persone infette, passano nello stato infected. Mentre nell'ultima fase, le persone infette dopo la guarigione transitano nello stato susceptible.

Per i prossimi due modelli, valgono le seguenti premesse:

- La diffusione di informazioni viene modellata attraverso un grafo $G = (V, E)$;
- Un nodo è chiamato *attivo* se ha adottato l'informazione;
- I nodi possono passare da uno stato inattivo ad uno stato attivo, ma non può succedere il contrario;
- La diffusione dell'informazione attraverso la rete $G$ è rappresentata come diffusione di nodi attivi;
- Dato un insieme iniziale $A$ di nodi attivi, si suppone che i nodi in $A$ siano i primi ad essere attivi, mentre i rimanenti non lo sono al tempo/step 0;
- Il processo di diffusione dei nodi attivi avviene al tempo/step $\geq 0$.

### Linear Threshold Model

I modelli basati su *threshold* furono per primi proposti da Mark Granovetter [38] per modellare comportamenti collettivi, con lo scopo di trattare problemi di decisione binari, come diffusione di innovazioni, dei *rumors*, malattie e così via. Nel *Linear Threshold model* ogni nodo $v \in V$ ha associato un peso $w_{u,v} > 0$ per ogni nodo vicino $u$ che ha un arco incidente in $v$, in modo tale che:

$$\sum_{u \ vicini \ di \ v} w_{u,v} \leq 1$$

Inoltre, per ogni nodo $v \in V$ una threshold $\theta_v$ viene scelta in maniera casuale appartenente all'intervallo $[0, 1]$. Dato un insieme di nodi attivi iniziale $A$, il processo di diffusione inizia da tali nodi. Ad ogni step $t$, un nodo non attivo $v$ viene influenzato da ognuno dei suoi nodi vicini attivi in base al peso $w_{u,v}$. Se il peso totale dei nodi vicini a $v$ è maggiore della soglia $\theta_v$, allora il nodo $v$ diventerà attivo nello step $t+1$. Il processo termina fintanto che non sono possibili ulteriori attivazioni [39].

### Independent Cascade Model

Grazie ai lavori descritti in [40,41] e dalla teoria delle probabilità, i *dynamic cascade models* sono stati considerati per i processi di diffusione. Nel contesto del marketing, *Goldenberg et al.* [42,43] furono i primi a studiare i *cascade models*. Nell'independent cascade model, il processo inizia da un insieme di nodi $A$. Ogni nodo $v$ che diventa attivo nello step $t$ ha un'unica possibilità di attivare ognuno dei suoi vicini attualmente non attivi. L'attivazione da parte del nodo $v$ di un nodo vicino $u$ (per cui esiste un arco da $v$ a $u$) avviene con probabilità pari al peso dell'arco $w_{v,u}$. Se esistono più nodi vicini che provano ad attivare $w$, questi proveranno in maniera sequenziale con ordine arbitrario. Se uno di questi tentativi ha successo, $w$ diventa attivo allo step $t+1$. Da sottolineare che ad ogni nodo è data una sola possibilità di attivare i propri vicini, per cui se fallisce al tempo $t$, non gli è permesso attivare i suoi vicini al tempo $t+1$. Così come il Linear Threshold model, il processo in tale modello termina quando non è possibile attivare nuovi nodi [44].

**Scelta del modello di diffusione** I primi tre modelli analizzati: *SI, SIR* e *SIS*, generalmente vengono utilizzati per modellare processi di diffusioni di malattie e non seguono quelle che sono le dinamiche di cascading behaviour. I modelli che più si adattano per la simulazione del processo di diffusione di false informazioni nelle reti sociali sono il *Linear Threshold Model* e l'*Independent Cascade Model*. Il primo necessita, oltre ad informazioni relative alle probabilità di spreading di false informazioni, ovvero i pesi sugli archi, anche che sia assegnata una threshold ad ogni nodo, a differenza dell'altro modello. Per questo motivo la scelta del modello di diffusione per la simulazione del processo di diffusione di false informazioni è ricaduta sull'**Independent Cascade Model** ed inoltre perché è lo stesso modello utilizzato dall'algoritmo (vedi 4), il quale è stato usato per avere un confronto con le soluzioni proposte nel lavoro di tesi.

# Capitolo 4

# Sorgenti di false informazioni: chi sospettare?

In questo capitolo viene descritto l'algoritmo **Imeter-Sort**, tratto dal lavoro di *Dung T. Nguyen, Nam P. Nguyen* e *My T. Thai* [45], il quale è stato implementato per poter effettuare dei confronti con le soluzioni proposte in questo lavoro di tesi.

## 4.1 L'algoritmo Imeter-Sort

Gli autori dell'algoritmo affrontano il problema di identificare le sorgenti di false informazioni studiando il problema *k-Suspector*, che mira a trovare i $k$ utenti più sospetti da un insieme di vittime, le quali sono già state influenzate dalla *misinformation*.
Una definizione più dettagliata del k-Suspector problem:
*Dati un grafo diretto e pesato $G = (V, E, p)$ che modella una rete sociale, in cui $p$ rappresenta il fatto che i pesi degli archi corrispondono a probabilità, l'Independent Cascade come modello di diffusione ed un insieme di nodi infetti $I$, lo scopo è quello di trovare un insieme $S$ di $k$ nodi sospetti per cui il numero di sorgenti della misinformation originale in $S$ è massimizzato.*

Di seguito vengono descritte le fasi che compongono l'algortimo Imeter-Sort: *reverse diffusion process* e *ranking*.

### 4.1.1 Reverse Diffusion Process

La misinformation che inizia da un insieme di "attaccanti" iniziale $A$ può espandersi in una popolazione più grande, solamente sfruttando quelle che sono le connessioni a partire dai nodi attivi ai loro vicini. Per cui l'idea di base degli autori è quella di invertire il processo di diffusione andando a rintracciare le sorgenti della misinformation da quello che è l'insieme di nodi attivi $I$.
Tale intuizione è raffigurata in figura 4.1.

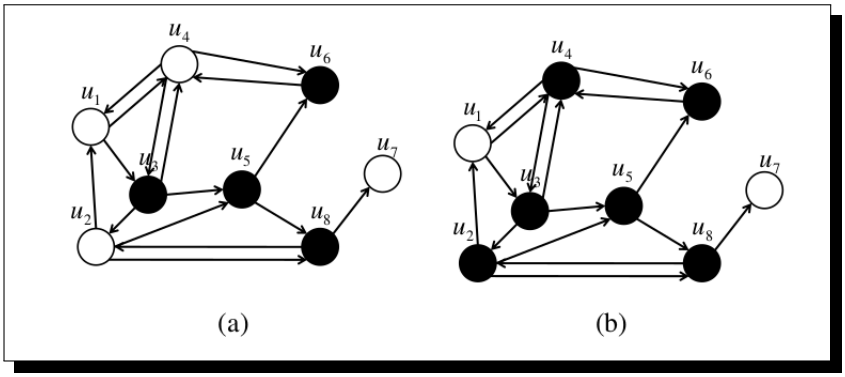


Figura 4.1: Due esempi di propagazione della misinformation.

Nella figura 4.1(a) è facile constatare che $u_3$ è la sorgente della misinformation che ha inizialmente influenzato $u_5$ e successivamente $u_6$ e $u_8$. Per dare un'idea dell'algoritmo, dall'insieme di vittime $I = \{u_3, u_5, u_6, u_8\}$ si può scoprire la sorgente nel seguente modo: i nodi $u_6$ e $u_8$ condividono come vicino solamente $u_5$, per cui o erano degli attaccanti oppure sono stati influenzati da $u_5$. $u_5$ potrebbe essere una vittima di $u_3$ oppure essere esso stesso un attaccante. Mentre $u_3$ è sicuramente un attaccante, in quanto non è presente alcun nodo attivo che avrebbe potuto influenzarlo. Nelle situazioni reali, ovviamente, è molto più complicato ripercorrere il processo di diffusione, dal momento che ogni nodo potrebbe essere sia un attaccante che essere stato influenzato da uno dei suoi vicini, come mostrato in figura 4.1(b). Di seguito una descrizione più formale del processo.
Dato un grafo $G = (V, E, p)$ ed un insieme di nodi attivi $I \subseteq V$, il reverse diffusion process inizia con $|I|$ reverse flow a partire dai nodi attivi in $I$. Ad ogni step del processo, un reverse flow si ferma al nodo corrente $u$ oppure avanza verso uno dei nodi attivi e vicini di $u$. Dal momento che un vicino attivo $v \in N_u^- \cap I$ di $u$ può aver attivato $u$ con probabilità $p_{v,u}$, dove $N_u^-$ è l'insieme dei nodi che hanno un arco direzionato verso $u$, la probabilità

che nessun vicino abbia attivato $u$, ovvero che la probabilità che il flusso si fermi ad $u$ è:

$$P\big(stop(u)\big) = \prod_{v \in N_u^- \cap I} (1 - p_{v,u})$$

Se questo reverse flow non si ferma in $u$, può avanzare in maniera casuale ad uno dei nodi attivi e vicini di $u$ proporzionalmente all'influenza che hanno sul nodo $u$. Per cui, la probabilità che il flusso avanzi da $u$ a $v$ è:

$$P(u \to v) = \frac{p_{v,u}}{\sum_{z \in N_u^- \cap I} p_{z,u}}$$

Inoltre, quando due o più reverse flow si uniscono, questi vengono combinati per formare un nuovo reverse flow. Il reverse diffusion process termina quando non sono presenti reverse flow attivi.

### 4.1.2 Ranking

Una osservazione chiave del reverse diffusion process è che più volte un nodo viene attraversato da un reverse flow, più è probabile che sia un attaccante. Pertanto, gli autori dell'algoritmo introducono la metrica *Imeter*, per valutare il livello di sospetto di un nodo. L'idea alla base di tale metrica è quella di contare il numero atteso di volte in cui ogni nodo viene attraversato da un reverse flow. Per cui, più è alto il valore Imeter di un nodo e più sono alte le probabilità che sia un attaccante. Tale valore viene stimato dopo aver effettuato $R$ esecuzioni del reverse diffusion process sulla rete, dove $R$ corrisponde a $log|I|$. Successivamente vengono ordinati i nodi in base al loro valore Imeter ed i primi $k$ nodi risultanti vengono indicati come le sorgenti che hanno diffuso false informazioni. Di seguito una rappresentazione schematica dell'algoritmo 1.

**Algorithm 1** Imeter-Sort

**Input:** Un grafo $G = (V, E, p)$, l'insieme dei nodi infetti $I$ e $k$
**Output:** I $k$ nodi più sospetti
1: $R \longleftarrow$ # di simulazioni
2: Esegui $R$ simulazioni del Reverse Diffusion Process per computare per ogni nodo il valore Imeter
3: Ordina i nodi secondo $Imeter(u_1) \geq Imeter(u_1) \geq \ldots \geq Imeter(u_{|I|})$
4: **return** $k$ nodi $u_1, u_2, \ldots u_k$

# Capitolo 5

# Identificazione di una singola sorgente

Con l'avvento dei social media, lo scambio di informazioni ha avuto un incremento esponenziale, sia in termini di grandi quantità di dati che nella velocità di diffusione. Tale considerazione si applica anche alla diffusione di false informazioni, la quale ha un forte impatto sia dal punto di vista sociale che economico, come discusso nella sezione 3.1. Risulta importante identificare le sorgenti di false informazioni per diversi motivi:

- Le forze dell'ordine sono interessate nel conoscere i responsabili della diffusione di false informazioni con le quali hanno manipolato i prezzi del mercato di determinati prodotti, diffamato persone, e così via;
- Rendere i social media un canale sicuro e fidato per la pubblicazione di importanti contenuti;
- Fornire indicazioni preziose nella progettazione di strategie efficaci per campagne di contenimento di tali informazioni. Infatti, molte strategie dimostrano di essere più efficaci quando si conoscono le sorgenti [26, 44, 46];
- Capire dal punto di vista sociale quali sono le motivazioni ed i meccanismi di tale fenomeno;
- ...

L'obiettivo di questo lavoro di tesi è quello di identificare le sorgenti di false informazioni all'interno di reti sociali. L'idea alla base di tale lavoro, a differenza dell'algoritmo descritto nel capito precedente (vedi 4), il quale prova a

percorrere in maniera inversa il processo di diffusione, è quella di capire come sia avvenuto tale processo, analizzando la struttura della rete sociale. A tal fine la rete sociale non viene considerata nella sua totalità, ma solamente una parte relativa ai nodi che sono stati "contagiati" dalla falsa informazione. Per cui, se si rappresenta la rete sociale come un grafo diretto e pesato $G = (V, E, p)$, la porzione di rete da analizzare corrisponde al sottografo indotto $H = (V', E', p)$, in cui l'insieme $V' \subseteq V$ è composto dai nodi che sono stati influenzati dalla falsa informazione ed $E' \subseteq E$ è costituito dagli archi $(u, v)$, dove sia $u$ che $v$ appartengono a $V'$. Inoltre, ad ogni arco $(u, v)$ è associato un peso che indica la probabilità che il nodo $v$ sia influenzato da $u$. Per cui, a partire dal sottografo indotto $H$, l'idea è quella di comprendere quali nodi abbiano dato vita a tale struttura, trovando lo spanning tree radicato in uno o più nodi avente probabilità massima, il quale dovrebbe rappresentare al meglio quello che è stato il processo di diffusione reale. In questo modo, le radici dello spanning tree calcolato vengono indicate come le sorgenti della falsa informazione. Di conseguenza con tale approccio il problema dell'identificazione delle sorgenti di false informazioni si riduce al problema di calcolare lo spanning tree con probabilità massima.

In questo capitolo viene analizzato il caso in cui è presente un'unica sorgente di false informazioni.
In generale, si parla di spanning tree quando si ha a che fare con grafi non diretti, mentre in questo scenario la rete sociale viene rappresentata mediante un grafo diretto. L'equivalente struttura per il caso diretto viene denominata **spanning arborescence**. Come descritto in 2.2.2, un'arborescence è un sottografo diretto $G = (V, E)$, il quale contiene esattamente un unico cammino diretto da un nodo specifico $u$ ad ogni altro vertice $v \in V'$, dove $V' \subseteq V$. Il nodo $u$ viene chiamato *radice*, in quanto non ha nessun arco entrante. Un'arborescence è una *spanning arborescence* se l'insieme dei vertici $V'$ è pari a $V$. In particolare, si è interessati a calcolare la spanning arborescence di peso massimo, dove per peso massimo si intende che la somma delle probabilità sia la massima possibile.
Per esplicitare visivamente i concetti descritti, in figura 5.1 viene mostrato un grafo d'esempio (fig. 5.1(a)), affiancato da una possibile spanning arborescence (fig. 5.1(b)) e dalla massima spanning arborescence (fig. 5.1(c)) aventi entrambe come radice il nodo 1.

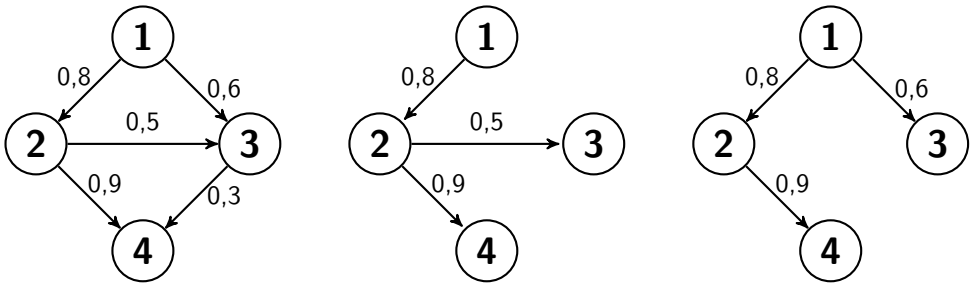


Figura 5.1: (a) Grafo d'esempio; (b) Una spanning arborescence; (c) La spanning arborescence di peso massimo

In letteratura, per il problema comunemente noto come *maximum spanning arborescence problem*, esistono diversi algoritmi. Uno dei più noti, se non il principale è l'algoritmo di **Chu-Liu/Edmonds**, il quale fu proposto in maniera indipendente da *Yoeng-Jin Chu* e *Tseng-Hong Liu* nel 1965 e successivamente da *Jack Edmonds* nel 1967. In questo lavoro di tesi tale algoritmo viene utilizzato per trovare la sorgente di false informazioni. Per cui, di seguito viene descritto in maniera dettagliata.

## 5.1 Algoritmo di Chu-Liu/Edmonds

Nella teoria dei grafi l'algoritmo di Edmonds, chiamato anche algoritmo di Chu-Liu-Edmonds, è utilizzato per determinare, a partire da un dato grafo pesato e fortemente connesso $G = (V, E, p)$, una spanning arborescence di peso massimo. L'algoritmo individua cioè un sottoinsieme degli archi del grafo $G$ che costituisce un albero avente come radice un nodo $u$ per cui esiste un cammino da $u$ ad ogni altro nodo e tale che il peso totale degli archi individuati risulti massimo. L'algoritmo si suddivide in una fase di *contrazione* ed una di *espansione*. Per tali fasi vengono utilizzate diverse strutture intermedie:

- Grafi $G^i$, che mantengono lo stato della rete;
- Grafi $B^i$, che conservano la soluzione corrispondente ai grafi intermedi $G^i$;
- Un *Merge-Find Set* $D$, comunemente noto come *union-find*, per tenere traccia dei nodi aggiunti alla soluzione corrente $B^i$;

- Liste di archi $Q^i$, che memorizzano eventuali cicli.

In generale, nella prima fase il grafo corrente $G^i$ viene contratto in un unico nodo per eliminare eventuali cicli, mentre nella seconda viene espanso per far riemergere i nodi del ciclo che erano stati contratti.

### Fase di contrazione

Come indica il nome, nella fase di contrazione il grafo di partenza $G$ viene contratto ogni qualvolta viene identificato un ciclo, fin quando alla fine si ottiene una soluzione, ovvero una spanning arborescence sul grafo risultante da tale processo. Una descrizione schematica e dettagliata di questa fase è presente in 2.

**Algorithm 2** Fase di contrazione

**Input:** Un grafo $G = (V, E, p)$
**Output:** Una spanning arborescence $B^i$ di $G^i$

$G^i = G$, $B^i = \phi$, $D = \phi$ ed $i = 0$
Finché esiste un nodo $v \in G^i$ non ancora visitato:
  Se $v \notin D$:
    Aggiungi $v$ sia a $D$ che a $B^i$
    Seleziona l'arco $(u, v)$ di peso massimo, $best\_in\_edge[v]$
    Se $(u, v)$ non genera un ciclo in $G^i$:
      Aggiungi $(u, v)$ a $B^i$ ed effettua *union-set* su $u$ e $v$
    Altrimenti:
      Salva gli archi del ciclo in $Q^i$ e quello di peso minimo $min\_edge^i$
      Incrementa il numero di livelli $i$ di 1
      Memorizza $G^i$ e $B^i$, poni $G^i = G^{i-1}$ e $B^i = B^{i-1}$
      Crea un nuovo nodo $w$ e contrai i nodi del ciclo in $w$
      Per ogni arco $(a, b)$ dove $a \in Q^i$:
        Aggiungi a $G^i$ l'arco $(w, b)$ di peso $p_{a,b}$
      Per ogni arco $(a, b)$ dove $b \in Q^i$:
        Aggiungi a $G^i$ l'arco $(a, w)$ di peso $p_{a,b} + p_{best\_in\_edge[b]} - p_{min\_edge^i}$
      Rimuovi da $G^i$ e $B^i$ gli archi di $Q^{i-1}$
      Effettua $D = D - \{v \in V \cap Q^{i-1}\}$
**return** $G^i$, $B^i$, $Q^i$ e $min_edge^i$ per ogni livello $i$

### Fase di espansione

La fase di espansione inizia a partire dalle strutture $G^i$ e $B^i$ calcolate nell'ultimo step della fase di contrazione. A partire da tali strutture e da quelle che contengono i cicli incontrati, si vanno ad eliminare quegli archi che riescono a rompere i cicli analizzati, per poi selezionare gli archi che apparterranno alla soluzione finale.
Una descrizione schematica e dettagliata di questa fase è presente in 3.

**Algorithm 3** Fase di espansione

**Input:** Le strutture $G^i$, $B^i$, $Q^i$, $min\_edge^i$ per tutti i livelli $i$
**Output:** Una spanning arborescence di peso massimo per $G$

Sia *edges* inizialmente pari agli archi di $B^i$
Finché $i \geq 0$:
  Sia $v$ il nodo risultante nella contrazione del ciclo salvato in $Q^{i-1}$
  Verifica che $v$ sia una radice nel grafo $G^i$ con vincoli su *edges*
  Aggiorna *edges* con gli archi presenti in $Q^i$
  Se $v$ è una radice in $G^i$:
    Rimuovi da *edges* l'arco memorizzato in $min\_edge^i$
  Altrimenti è presente un arco incidente a $v$:
    Trova tale arco nel grafo $G_{i-1}$ ed eliminalo da *edges*
  Decrementa $i$ di 1
**return** *edges*

Il risultato di tale fase è la spanning arborescence di peso massimo. La correttezza dell'algoritmo viene dimostrata in [47] e non viene analizzata in questo lavoro, ma l'idea alla base è che per ogni nodo venga selezionato l'arco incidente di peso massimo e che i cicli riscontrati vengano risolti eliminando gli archi di peso minimo.
Per cui, come specificato in precedenza, la radice della spanning arborescence viene indicata come la sorgente che ha diffuso false informazioni all'interno della rete sociale. Esperimenti e risultati in merito alla soluzione proposta vengono mostrati in 7.3.1.

# Capitolo 6

# Individuazione di sorgenti multiple

Nel capitolo precedente è stato considerato il caso in cui vi fosse un'unica sorgente che avesse diffuso false informazioni. La naturale evoluzione del problema è quella di identificare sorgenti multiple. A tal fine viene utilizzato lo stesso principio alla base del caso più semplice, in cui invece di calcolare una singola spanning arborescence, si determinano una o più *branching* di peso massimo. Come detto in precedenza in 2.2.2, una branching $B$ di $G = (V, E)$, con $G$ grafo diretto, è una foresta di arborescence disgiunte. Per cui, in una branching possono essere presenti più nodi senza archi entranti, i quali saranno indicati come sorgenti di false informazioni.
In letteratura scientifica un algoritmo per tale problema è stato pubblicato da *Camerini*, *Fratta* e *Maffioli* nel 1980 [48]. La metodologia proposta da Camerini et al., descritta successivamente, viene utilizzata per quello che è lo scopo di questo lavoro di tesi, sebbene non nella sua interezza come verrà discusso nella sezione 6.2.

## 6.1 Algoritmo di Camerini, Fratta e Mattioli

In questa sezione viene descritto l'algoritmo per il calcolo delle $k$ best branching. Il principio alla base è che le $k$ branching di peso massimo differiscono l'una dall'altra solamente di un arco. Per cui, in primo luogo, gli autori propongono di calcolare la branching di peso massimo 6.1.1. In quest'ultima esiste un arco $e$ che non dovrebbe appartenere alla successiva branching 6.1.2. A questo punto la terza branching potrà avere o meno l'arco $e$ nel

suo insieme degli archi e così via per le successive 6.1.3. L'algoritmo viene descritto nelle sezioni successive in base a quelle che sono le sue fasi.

### 6.1.1 Calcolo della branching di peso massimo

L'algoritmo per il calcolo della branching migliore è simile a quello analizzato in 5.1, ma differisce per alcuni aspetti. Di seguito vengono introdotte alcune definizioni e strutture per descrivere l'algoritmo.
Sia $N_{Y,Z} = (V, E, \sigma, \tau, \omega)$ una rete, dove:

- $V = 1, 2, \ldots, n$ è l'insieme dei vertici;
- $E = 1, 2, \ldots, m$ è l'insieme di archi di $N$;
- $\sigma$ e $\tau$ sono le *funzioni d'incidenza*, in cui $\sigma(e) = u$ e $\tau(e) = v$ indicano che l'arco $e$ è diretto dal vertice $u$ al nodo $v$;
- $\omega : E \to \mathbb{R}$ è la funzione peso degli archi.
- $Y$ è una qualsiasi branching di $N$, i cui archi devono far parte della soluzione (opzionale)
- $Z$ è l'insieme di archi che non devono far parte della soluzione (opzionale)

Inoltre nell'insieme dei vertici è presente un nodo fittizio, *root*, e nell'insieme degli archi un arco diretto da *root* ad ogni altro vertice $v \in V$ con peso pari a $-\infty$. L'algoritmo prevede una fase di contrazione della rete, nel momento in cui vengono riscontrati dei cicli. Supponendo di aver rilevato un ciclo su un sottoinsieme di nodi $V'$, $C = (V' = \{v_1, v_2, \ldots, v_k = v_1\}, E' = \{e_1, e_2, \ldots, e_{k-1}\}, \sigma, \tau, \omega)$, il quale non contiene il nodo *root* e tale che per ogni $h$, con $1 \leq h \leq k$, $\omega(e_h)$, l'arco $e_h$ è il massimo fra tutti gli archi diretti in $v_{h+1}$. Una rete $N_C = (V_C, E_C, \sigma_C, \tau_C, \omega_V)$ viene denominata *rete $N$ collassata in $C$*, ed è definita come segue. Siano $u_1 = root, u_2, \ldots, u_{n-k+1}$ i vertici di $V - V'$ e $V_C = \{u_1, u_2, \ldots, u_{n-k+2}\}$. Dove $u_{n-k+2}$ è un nuovo vertice (supernodo), ovvero il vertice di $N_C$ corrispondente a $C$.
Sia $E_C = E - \{e \in E : \sigma(e), \tau(e) \in V'\}$, per ogni arco $e \in E_C$:

$$\sigma_C(e) = \begin{cases} u_{n-k+2} & se\ \sigma(e) \in V' \\ \sigma(e) & altrimenti, \end{cases}$$

$$\tau_C(e) = \begin{cases} u_{n-k+2} & se\ \tau(e) \in V' \\ \tau(e) & altrimenti, \end{cases}$$

$$\omega_C(e) = \begin{cases} \omega(e) - \omega(e_h) & se\ \tau(e) \in V' \\ \omega(e) & altrimenti, \end{cases}$$

dove $e_h$ è l'unico arco di $C$ che collide con $e$, ovvero per cui $\tau(e_h) = \tau(e)$. Di conseguenza, questo è il modo di contrarre ed aggiornare la rete in presenza di cicli. Per descrivere l'algoritmo, di seguito vengono introdotte le principali strutture utilizzate:

- Una branching $T$ contenente le relazioni fra i nodi in presenza di cicli;
- Una branching $B$ che memorizza la soluzione corrente per la rete;
- Un dizionario $\beta$ che tiene traccia per ogni nodo dell'arco, selezionato in $B$, incidente su tale nodo.

Una descrizione dell'algoritmo è mostrata in 4.

**Algorithm 4** Best(Y, Z)

**Input:** Una rete $M$
**Output:** Una branching di peso massimo di $M$

Siano $T$ una branching contenente i vertici isolati di $M$ e $B = \phi$
Finchè esiste un nodo esposto $v \neq root$ rispetto a $B$:
  Sia $b$ arco tale che $\tau_M(b) = v$ con $\omega_M(b)$ di peso massimo
  Aggiungi $b$ a $B$ e poni $\beta(v) = b$
  Se $B$ contiene un ciclo $C$:
    Collassa la rete $M$ in $M_C$
    Sia $u$ il vertice di $M_C$ corrispondente a $C$, aggiungi $u$ a $T$
    Per ogni vertice $w$ di $C$ aggiungi l'arco $(u, w)$ a $T$
    Elimina da $B$ gli archi in cui i nodi di $C$ son coinvolti
  Finché $T$ contiene nodi isolati non analizzati:
    Identifica in $T$ un percorso di vertici $v_1, v_2, \ldots, v_k$
    Sia $v_1$ è una radice non isolata in $T$ e $v_k = \tau(\beta(v_1))$
    Per $h$ che assume valori da 1 a $k - 1$
      Assegna il valore contenuto in $\beta(v_h)$ a $\beta(v_{h+1})$
      Rimuovi da $T$ il nodo $v_h$ e tutti gli archi uscenti da esso
Sia $A = \{\beta(v) : v \neq$ vertice di $N\}$
Elimina da $A$ il nodo $root$ e tutti gli archi in cui è coinvolto
**return** $A$

### 6.1.2 Ricerca della branching successiva

In questa sezione viene descritta la procedura che identifica la branching successiva, a partire dalla branching di peso massimo. Gli autori dell'algoritmo dimostrano nella loro pubblicazione che le migliori branching differiscono solamente di un arco ed inoltre forniscono una metodologia per ricercare questo arco.
Sia $\bar{A}$ il risultato dell'esecuzione dell'algoritmo *Best* 4, per ogni arco $b \in \bar{A} - Y$ e siano $\sigma_M, \tau_M, \omega_M$ le rispettive funzioni di incidenza e peso della rete corrente $M$ considerata in *Best*, quando $b$ è stato scelto in linea 3. Un arco $f$ di $M$ viene chiamato *successivo* a $b$ se soddisfa le seguenti proprietà:

- $\tau_M(f) = \tau_M(b)$ con $f \neq b$;
- Non esiste un percorso da $\tau(b)$ a $\sigma(f)$ in $\bar{A}$;
- $\omega_M(f)$ è massimo fra tutti gli archi che soddisfano le proprietà precedenti.

Nel caso in cui tale arco $f$ esiste, si pone $\delta(b) = \omega_M(b) - \omega_M(f)$, altrimenti pari a 0. A questo punto sia

$$d = \min_{b \in \bar{A} - Y} \delta(b),$$

e se $d \leq +\infty$ l'arco $e \in \bar{A} - Y$ è l'arco per cui $\delta(e) = d$. Se la rete $N_{YZ}$ non ha una branching successiva allora $d = +\infty$, altrimenti $A' = Best(Y, Z \cup \{e\})$ è la branching successiva di $N_{YZ}$ e $\omega(\bar{A}) - \omega(A') = d$. Per cui, data $\bar{A}$, una variante di $Best(Y, Z)$ viene eseguita, chiamata $Next$ 5 risolvendo in linea 3 gli archi selezionati in favore di $\bar{A}$ laddove possibile. Ogniqualvolta un arco $b \in \bar{A} - Y$ viene scelto, un arco $f$ successivo a $b$ viene ricercato 6 e $\delta(b)$ viene ottenuto. Quindi è possibile computare sia $e$ che $d$, e come detto in precedenza, se $d \leq +\infty$ allora $\omega(\bar{A}) - d$ è il peso di $A'$ e $Best(Y, Z \cup \{e\})$ porta alla branching successiva.

**Algorithm 5** Next(A, Y, Z)

**Input:** Una rete $N_{YZ}$ ed una branching di peso massimo $A$

1: Siano $M = N_{YZ}$, $d = +\infty$ e $B = \phi$
2: Finchè esiste un nodo esposto $v \neq root$ rispetto a $B$:
3: Sia $b$ arco tale che $\tau_M(b) = v$ con $\omega_M(b)$ di peso massimo
4: Laddove possibile considerare l'arco $b \in A$
5: Aggiungi $b$ a $B$ e poni $\beta(v) = b$
6: Se $b \in A - Y$:
7: $f = Seek(b, A)$
8: Se $\omega_M(b) - \omega_M(f) < d$:
9: $e = b$
10: $d = \omega_M(b) - \omega_M(f)$
11: Se $B$ contiene un ciclo $C$:
12: Collassa la rete $M$ in $M_C$
13: Elimina da $B$ gli archi in cui i nodi di $C$ son coinvolti
14: **return** $e, d$

Nella procedura $Seek$, la coda a priorità $Q$ contiene gli archi incidenti a $\tau(b)$ ordinati per peso massimo. L'operazione $MAX(Q)$ permette di ottenere un arco $f'$, il quale deve soddisfare le proprietà elencate in precedenza per essere l'arco successivo a $b$. Nel caso in cui tale arco non venga trovato, viene restituito un arco fittizio, etichettato nell'algoritmo con 0.

**Algorithm 6** Seek(b, A)

1: Sia $Q$ la coda a priorità utilizzata per trovare $b$ (vedi linea 3 di Next)
2: Finchè $Q$ non è vuota:
3: $f = MAX(Q)$
4: Se non esiste un cammino da $\tau(b)$ a $\sigma(f)$:
5: Aggiungi $f$ a $Q$
6: Restituisci $f$
7: $\omega_M(0) = -\infty$
8: Restituisci 0

### 6.1.3 Determinazione delle k migliori branching

Nelle sezioni precedenti sono stati descritti gli strumenti necessari per il calcolo delle $k$ branching di peso massimo, mentre in questa viene descritto un algoritmo per classificarle.
Siano $A^{(i)}$, per $1 \leq i \leq k$, le $k$ branching in ordine di peso non crescente. Supponendo che l'algoritmo di classificazione ha come output $A^{(1)}, A^{(2)}, \ldots, A^{(j-1)}$ per $j > 1$, le rimanenti branching vengono partizionate in $j > 1$ insiemi disgiunti della forma:

$$P_i^{(j-1)} = \{A^{(h)} : h > j-1; Y_i^{(j-1)} \subseteq A^{(h)} \subseteq E - Z_i^{(j-1)}\},$$

per $1 \leq i \leq j$. $A^{(i)}$ soddisfa tutte le condizioni di questi insiemi tranne la prima, $h \geq j-1$. Una branching in questo insieme è identificata da una branching *successiva* $A'$ che soddisfa:

$$Y_i^{(j-1)} \subseteq A' \subseteq E - Z_i^{(j-1)}.$$

L'algorimo 7 utilizza una lista $P$ per rappresentare la partizione. Ogni insieme $P_i^{(j-1)}$ è rappresentato da una tupla $(w, e, A, Y, Z)$ in $P$, dove $A = A^{(i)}$, $Y = Y_i^{(j-1)}$, $Z = Z_i^{(i)}$, $w$ è il peso associato alla branching in $P_i^{(j-1)}$, ovvero $A'$, e l'arco $e$ di $A$ identifica $A'$, la quale è il risultato di $Best(Y, Z \cup \{e\})$.
Nell'algoritmo viene selezionata la tupla avente $w$ massimo, viene eseguita $Best(Y, Z \cup \{e\})$ per ottenere la $A^{(j)}$ e viene formata una nuova partizione $P_h^{(j)}$ per $1 \leq h \leq j$, suddividendo $P_i^{(j)}$. Nell'ultima operazione $P_i^{(j)} - \{A^{(j)}\}$ viene partizionato in due sottoinsiemi: uno composto dalla branching contenente l'arco $e$ della tupla selezionata e l'altro senza $e$. Successivamente viene invocata la procedura $Next$, ovvero $Next(A^{(i)}, Y \cup \{e\}, Z)$ e $Next(A^{(j)}, Y, Z \cup \{e\})$.

**Algorithm 7** Ranking(N, k)

**Input:** Una rete $N$ ed un numero intero $k$
**Output:** Le $k$ branching di peso massimo di $N$

$A^{(1)} = Best(\phi, \phi)$
$e, d = Next(A^{(1)}, \phi, \phi)$
Aggiungi a $P$ la tupla $(w(A^{(1)}) - d, e, A^{(1)}, \phi, \phi)$
Per $j$ che assume valori da 2 a $k$:
    Rimuovi da $P$ la tupla $P_i^{(j-1)} = (w, e, A, Y, Z)$ per cui $w$ è massimo
    Se $w = -\infty$:
     Restituisci le branching calcolate, perchè non ne sono presenti altre
    $Y' = Y \cup \{e\}$ e $Z' = Z \cup \{e\}$
    $A^{(j)} = Best(Y, Z')$
    $e, d = Next(A, Y', Z)$
    Aggiungi a $P$ la tupla $(w(A) - d, e, A, Y', Z)$
    $e, d = Next(A^{(j)}, Y, Z')$
    Aggiungi a $P$ la tupla $(w - d, e, A^{(j)}, Y, Z')$

## 6.2 Euristiche applicate

L'algoritmo di *Camerini et al.* analizzato in 6.1 consente di calcolare, dato un numero intero positivo $k$, le $k$ branching di peso massimo. Per risolvere il problema dell'identificazione delle sorgenti multiple che avessero diffuso false informazioni, l'idea era quella di utilizzare l'algoritmo descritto precedentemente, in modo da indicare le radici delle $k$ branching come le sorgenti di false informazioni. Tuttavia da esperimenti effettuati, si è notato che tale algoritmo non è ideale per il problema, in quanto le $k$ branching presentavano sempre le stesse radici. Poiché, sostanzialmente, le $k$ branching differiscono soltanto di pochi archi fra di loro, ciò non è sufficiente a far variare le radici. Per questo motivo si è dovuta trovare una soluzione differente. A tal fine è stato preso in considerazione l'algoritmo 4 come punto di partenza, in quanto in grado di fornire un numero di sorgenti $j \leq k$. Per le restanti $k - j$ sorgenti, sono state analizzate principalmente due euristiche:

1. Date le $j$ sorgenti $\{v_1, \ldots, v_j\}$, sono stati modificati i pesi associati agli archi $(v, w)$, con $v \in \{v_1, \ldots, v_j\}$ e $w$ appartenente all'insieme dei vertici della rete sociale, fissando:
   (a) $\omega_{(v,w)} = 0$;
   (b) $\omega_{(v,w)} = -\infty$;

2. Date le $j$ sorgenti $\{v_1, \ldots, v_j\}$ eliminare tutte le $j$ sorgenti e gli archi in cui sono coinvolte.

Per ognuna di esse una volta applicata l'euristica, l'algoritmo *Best* 4 è stato eseguito su queste nuove configurazione della rete. Tale processo è stato ripetuto finchè il numero di sorgenti calcolate non avesse raggiunto il valore $k$. Da diversi esperimenti la prima euristica è risultata meno efficiente, in quanto per entrambe le variazioni dei pesi associati agli archi, dall'algoritmo *Best* non venivano identificate nuove sorgenti rispetto a quelle calcolate in precedenza. Mentre la seconda euristica è risultata migliore, permettendo di trovare nuove sorgenti. Di conseguenza tale euristica è stata utilizzata per fornire una soluzione al problema originale. Nella sezione 7.3.2 vengono presentati gli esperimenti effettuati ed i risultati di tale approccio.

# Capitolo 7

# Esperimenti e risultati

Le fasi di analisi degli algoritmi descritti nei precedenti capitoli hanno richiesto lo svolgimento, in parallelo, di una fase di implementazione e testing degli stessi. Un approccio di questo tipo ha permesso di poter sviluppare gli algoritmi, testando le componenti che li compongono separatamente, in modo da capire se le scelte effettuate risolvessero i vari problemi in maniera corretta. Si analizzano ora in dettaglio le fasi di implementazione e testing.

## 7.1 Implementazione

Per quanto riguarda la fase di implementazione, gli algoritmi e le procedure descritte nei capitoli precedenti sono state scritte utilizzando il linguaggio di programmazione **Python**.

Python è un linguaggio di programmazione ad alto livello, rilasciato pubblicamente per la prima volta nel 1991 dal suo creatore *Guido van Rossum*. Tale linguaggio supporta diversi paradigmi di programmazione, come ad esempio *object-oriented*, *imperativo* e *funzionale* ed è provvisto di una libreria *built-in* estremamente ricca che, insieme ad ulteriori caratteristiche, rendono Python uno dei linguaggi più comodi da utilizzare oggigiorno. Tale linguaggio è *pseudocompilato*: vi è un interprete che analizza ed esegue il codice sorgente. Python è un linguaggio *portabile*: una volta scritto il sorgente, ne è consentito l'utilizzo in un ambiente di esecuzione diverso da quello originale. Un'altra caratteristica fondamentale di questo linguaggio è quella di essere *free software*: l'interprete e le librerie sono gratuite ed il linguaggio può essere modificato e ridistribuito in accordo ad una licenza *open source*. Numerosi sono i vantaggi e le caratteristiche di Python (per

maggiori dettagli ci si può riferire a [49, 50]).

La versione utilizzata per tale lavoro è la 2.7 di Python. Inoltre, è stato necessario l'utilizzo di alcune funzioni *built-in* e della libreria *NetworkX* [51].

**NetworkX** è una libreria utile per la creazione, manipolazione e per lo studio di dinamiche in reti complesse. Consente di memorizzare le reti in vari formati, generare varie tipologie di reti classiche e random, mostrare una rappresentazione grafica di una rete, costruire modelli e molto altro.

Una descrizione completa del codice sviluppato è fornita in [52]. La fase di implementazione ha previsto la realizzazione delle seguenti componenti:

- *Modello di diffusione.* È stato realizzato il codice relativo al modello di diffusione *Independent Cascade* descritto in 3.2. La procedura prende in input un insieme di nodi che rappresentano le sorgenti di false informazioni ed in base alla struttura del grafo simula un processo di diffusione, restituendo i nodi che sono stati influenzati;

- *Sottografo degli infetti.* A partire dall'insieme dei nodi influenzati dalla falsa informazione, risultanti dal processo di diffusione sul grafo originale, si è estratto il sottografo indotto (vedi 2.2.2). Ovvero il grafo constituito dai nodi "infetti" e dagli archi fra di essi;

- *Algoritmo Imeter-Sort.* È stato implementato l'algoritmo proposto da *Dung T. Nguyen, Nam P. Nguyen* e *My T. Thai* basandosi sullo pseudocodice (vedi algoritmo 1) presente nella loro pubblicazione [45], facendo alcune assunzioni laddove non fossero spiegate in dettaglio le scelte effettuate. Tale implementazione è stata sviluppata in modo da effettuare un confronto con le soluzioni proposte, a parità di sorgenti, probabilità sugli archi del grafo e sottografo indotto degli infetti;

- *Algoritmo di Chu-Liu/Edmonds.* È stato utilizzato per l'individuazione di una singola sorgente di false informazioni. Un'implementazione dell'algoritmo fedele alla pubblicazione di *Edmonds* [47] è disponibile grazie alla libreria NetworkX. Tuttavia per quest'ultima sono state introdotte diverse ottimizzazioni (vedi 7.2);

- *Algoritmo di Camerini, Fratta e Mattioli.* Per il caso generale dell'identificazione di sorgenti multiple di false informazioni è stato sviluppato l'algoritmo nella sua interezza descritto dagli autori nella pubblicazione [48]. Come descritto in 6.2, sono state realizzate diverse euri-

stiche, basate su parte dell'algoritmo, per la risoluzione del problema originale.

La libreria NetworkX è stata di supporto per l'implementazione di entrambi gli algoritmi. Un primo utilizzo è stato quello relativo alla lettura del grafo da file. La funzione utilizzata è *read_edgelist* che prende in input molteplici parametri, di cui alcuni opzionali. Nel caso in questione, si utilizzano due parametri per la lettura: il primo indica il path relativo al file, mentre il secondo serve per far capire il tipo di grafo da costruire (in questo caso diretto). I grafi, utilizzati per questo lavoro, sono rappresentati da un file contenente la lista di archi del grafo. Un esempio reale di grafo utilizzato è mostrato in Figura 7.1.

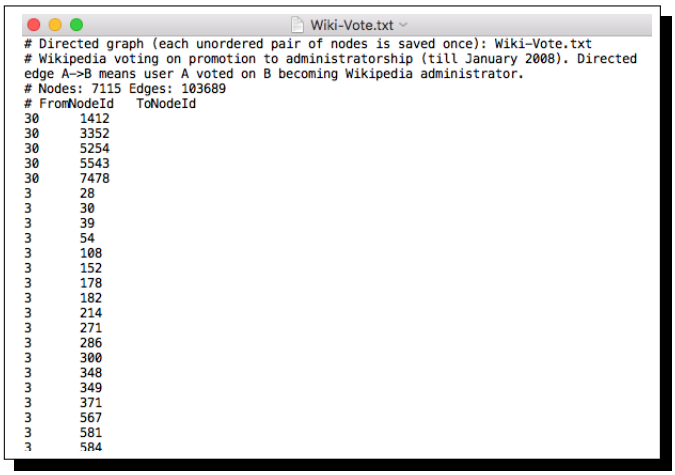


Figura 7.1: Porzione del file Wiki-Vote.txt contenente la lista di archi del grafo.

Oltre alla lettura del grafo, di seguito si descrivono le principali procedure o classi di NetworkX utilizzate per questo lavoro:

- *MultiDiGraph_EdgeKey*, classe che incapsula l'implementazione di multigrafo (vedi 2.2.2), il quale assegna una chiave univoca ad ogni arco, associandogli un dizionario che effettua il *mapping* tra le chiavi degli archi e le loro informazioni. Tale classe è stata utilizzata per la rappresentazione dei grafi sia per l'algoritmo di Edmonds che di Camerini;

- *Edmonds*, classe che fornisce i metodi per il calcolo di spanning arborescence di peso massimo o minimo. L'algoritmo implementato in questa classe è stato poi modificato, come descritto in 7.2;
- *UnionFind*, classe che implementa il *MergeFind Set*, utilizzata per tenere traccia dei nodi aggiunti alla soluzione corrente sia nell'algoritmo di Edmonds che di Camerini;
- *weakly_connected_component_subgraphs*, metodo impiegato per verificare la presenza di più sottografi, ed ottenere i nodi relativi alle singole componenti connesse, derivanti dal processo di diffusione;
- *shortest_path*, procedura utilizzata nell'implementazione dell'algoritmo di *Camerini et al.* per verificare l'esistenza di un cammino tra due nodi per la ricerca di una branching "successiva" in 6.

Inoltre sono state adoperate, oltre a NetworkX, altre due librerie Python:

- *psutil*, libreria *cross-platform* in grado di fornire informazioni sul sistema e sui processi in esecuzione [53], utilizzata per monitorare l'utilizzo di memoria per l'algoritmo di Edmonds;
- *multiprocessing*, *package* che offre concorrenza sia locale che remota, che permette di "aggirare" il *Global Interpreter Lock* di Python utilizzando sottoprocessi, invece che thread [54]. È stato utilizzato per l'esecuzione in parallelo dei test per l'individuazione di sorgenti multiple.

## 7.2 Ottimizzazioni

Nell'implementazione dei due algoritmi, sono state effettuate varie scelte atte a velocizzare il più possibile i tempi di esecuzione dei due algoritmi.

L'algoritmo di *Chu-Liu/Edmonds* descritto in 5.1 conserva tutte le strutture dati intermedie necessarie per l'esecuzioni. In particolare, i multigrafi $G^i$ e grafi $B^i$ identificati dal numero di livelli $i$, il quale sostanzialmente indica il numero di cicli riscontrati. L'algoritmo fornito dalla libreria NetworkX, così come descritto nel lavoro originale di Edmonds [47], salva tali strutture nella loro interezza. Un'ottimizzazione introdotta in questo lavoro di tesi consiste nel memorizzare solo le informazioni che differenziano i multigrafi $G^i$ ed i grafi $B^i$. Infatti, da un'analisi sull'algoritmo, si è notato che ogni

$G^i$ differisce dall'altro soltanto per gli archi ed i nodi relativi al $i$-esimo ciclo riscontrato. Quindi nella fase di contrazione, invece di conservare i grafi nella loro totalità, vengono memorizzati soltanto gli archi che hanno almeno un'estremità appartenente all'insieme dei vertici del ciclo. Questo processo viene effettuato per ogni ciclo riscontrato e non intacca la correttezza dell'algoritmo per la fase successiva.
Mentre per i grafi $B^i$ , si è osservato che nella fase di espansione, sostanzialmente sono necessari soltanti gli archi appartenenti al grafo $B^i$ per l'ultimo passo della fase di contrazione. Quindi, non vengono memorizzati tutti i grafi $B^i$ parziali, ma soltanto quello finale.
Da esperimenti effettuati tali ottimizzazioni hanno portato ad abbassare lo spazio in memoria utilizzato (seppure ancora elevato) ed il tempo di esecuzione dell'algoritmo. Inoltre tale modifica all'algoritmo di Edmonds è stata presentata agli autori della libreria di NetworkX e sto lavorando con gli stessi, in modo che possa essere integrata in una versione futura della libreria.

Mentre per quel che riguarda l'algoritmo di Camerini, Fratta e Mattioli descritto in 6.1 è stata modificata la struttura contenente le relazioni fra i nodi in presenza di cicli, denominata come $T$. Come descritto nell'algoritmo *Best* (vedi 4), ogniqualvolta viene riscontrato un ciclo $C$, viene creato un nuovo vertice $u$ e vengono aggiunti gli archi $(u, w)$ a $T$, per ogni vertice $w \in C$. La modifica a tale struttura introdotta nell'implementazione consiste nel creare il nuovo vertice $u$ e di aggiungere gli $(w, u)$ a $T$, per ogni vertice $w \in C$, in modo tale che risulti più facile identificare successivamente un percorso di vertici $v_1, v_2, \ldots, v_k$, in cui $v_1$ è una radice non isolata in $T$ e $v_k = \tau(\beta(v_1))$.

## 7.3 Testing

In questa sezione, saranno mostrati i confronti fra le soluzioni al problema dell'identificazione di sorgenti di false informazioni descritte precedentemente, mettendo in evidenza i risultati provenienti dalle varie esecuzioni. Inizialmente, viene descritto in che modo si è scelto di confrontare gli algoritmi. Successivamente, si mostrano i test effettuati al variare di alcuni parametri. La fase di testing ha consentito di mostrare l'efficacia delle soluzioni proposte. Nelle sezioni successive, si mostrano una serie di confronti che mettono in evidenza i risultati ottenuti. Nella prima sezione vengono confrontati l'algoritmo di Edmonds ed Imeter-Sort per l'identificazione della singola sorgente, mentre nella seconda la variazione all'algoritmo di Came-

rini ed Imeter-Sort. Per entrambi i confronti, sono stati considerati diversi grafi di partenza, a partire dai quali, per ogni esperimento, si è effettuata una simulazione del processo di diffusione. Successivamente a partire dal sottografo indotto risultante da tale processo, si è proceduto ad eseguire gli algoritmi. Quindi i parametri in gioco per i diversi esperimenti sono: i grafi di partenza, il numero di sorgenti e le sorgenti stesse da cui far partire il processo di diffusione e le taglie dei sottografi indotti derivanti dall'esecuzione dell'Independent Cascade Model.

### 7.3.1 Singola sorgente

### 7.3.2 Sorgenti multiple

# Capitolo 8

# Conclusioni e sviluppi futuri

# Bibliografia


[1] Scott John P. Social network analysis: A handbook (2nd edition). thousand oaks, ca: Sage publications.

[2] Università di Pisa Dipartimento di Scienze Sociale. http://sna.dss.unipi.it/Analisi%20delle%20reti.html.

[3] Tönnies Ferdinand. Gemeinschaft und gesellschaft, leipzig: Fues's verlag. (translated, 1957 by charles price loomis as community and society, east lansing: Michigan state university press).

[4] Durkheim Emile. De la division du travail social: étude sur l'organisation des sociétés supérieures, paris: F. alcan. (translated, 1964, by lewis a. coser as the division of labor in society, new york: Free press).

[5] Georg Simmel. Soziologie, leipzig: Duncker & humblot.

[6] Malinowski Bronislaw. The family among the australian aborigines: A sociological study. london: University of london press.

[7] Alfred Reginald Radcliffe-Brown. The social organization of australian tribes. sydney, australia: University of sydney oceania monographs, no.1.

[8] Alfred Reginald Radcliffe-Brown. On social structure. journal of the royal anthropological institute 70: 1–12. doi:10.2307/2844197.

[9] Lévi-Strauss Claude. Les structures élémentaires de la parenté. paris: La haye, mouton et co. (translated, 1969 by j. h. bell, j. r. von sturmer, and r. needham, 1969, as the elementary structures of kinship, boston: Beacon press).

[10] Nadel S. F. The theory of social structure. london: Cohen and west.

[11] Parsons Talcott. The structure of social action: A study in social theory with special reference to a group of european writers. new york: The free press.

[12] Parsons Talcott. The social system. new york: The free press.

[13] Blau Peter. Bureaucracy in modern society. new york: Random house, inc.

[14] Blau Peter. The american journal of sociology, (65)6: 545-556.

[15] Blau Peter. Exchange and power in social life.

[16] Bernie Hogan. TheNetworkedIndividual:AProfileofBarryWellman.

[17] Granovetter Mark. Introduction for the french reader. sociologica 2: 1–8.

[18] Wellman Barry. Structural analysis: From method and metaphor to theory and substance. pp. 19-61 in b. wellman and s. d. berkowitz (eds.) social structures: A network approach, cambridge, uk: Cambridge university press.

[19] Mullins Nicholas. Theories and theory groups in contemporary american sociology. new york: Harper and row, 1973.

[20] Tilly Charles. An urban world. boston: Little brown, 1974.

[21] Wellman Barry. Structural analysis: From method and metaphor to theory and substance. pp. 19-61 in social structures: A network approach, edited by barry wellman and s. d. berkowitz. cambridge: Cambridge university press.

[22] Jackson Matthew O. *Social and Economic Networks.*

[23] Papachristos Andrew. Murder by structure: Dominance relations and the social structure of gang homicide. american journal of sociology (the university of chicago press).

[24] Granovetter Mark. The impact of social structure on economic outcomes. the journal of economic perspectives.

[25] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World.*

[26] Amr El Abbadi Ceren Budak, Divyakant Agrawal. Limiting the spread of misinformation in social networks.

[27] Howell L. Digital wildfires in a hyperconnected world. in: Report 2013. world economic forum.

[28] Shaw DL Mccombs ME. The agenda-setting function of mass media. the public opinion quarterly 36: 176–187.

[29] Social Media Marketing Meets Web 2.0 and 55(3): 261-271. Creative Consumers: Implications for International Marketing Strategy. In Business Horizons, 2012. Berthon, p.r., pitt, l. f., plangger, k.

[30] Guest Editorial: How Technology is Changing the Design and 4(1): 1-5 Delivery of Services. In Operations Management Research, 2011. Davis, m.m., spohrer, j.c., maglio, p.p.

[31] Lord McAlpine to Demand Charity Donations for False Twitter Allegations. O'carroll l.

[32] Sweney M. Itv to pay lord mcalpine £125,000 damages.

[33] The Associated Press CBS News. Egypt newspaper fights cartoons with cartoons.

[34] Durden T. Fake tweets about syrian president assad's death cause all too real spike in crude and s&p. zerohedge.com.

[35] The Wall Street Journal. Fear sparks bangalore exodus.

[36] Biswas S. Social media and the india exodus. bbc world news.

[37] M. E. J. Newman. Networks: An introduction.

[38] M. S. Granovetter. Threshold models of collective behavior. the american journal of sociology, 83(6):1420–1443, 1978.

[39] Ryohei Nakano Masahiro Kimura, Kazumi Saito and Hiroshi Motoda. Extracting influential nodes on a social network for information diffusion.

[40] R. Durrett. Lecture notes on particle systems and percolation. wadsworth publishing, 1988.

[41] T. M. Liggett. Interacting particle systems. springer, 1985.

[42] B. Libai J. Goldenberg and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. marketing letters, 3(211-223), 2001.

[43] B. Libai J. Goldenberg and E. Muller. Using complex systems analysis to advance marketing theory development. academy of marketing science review, 2001.

[44] My T. Thai e Stephan Eidenbenz Nam P. Nguyen, Guanhua Yan. Containment of misinformation spread in online social networks.

[45] Nam P. Nguyen Dung T. Nguyen and My T. Thai. Sources of misinformation in online social networks: Who to suspect?

[46] D. T. Nguyen T. N. Dinh and M. T. Thai. Cheap, easy, and massively effective viral marketing in social networks: Truth or fiction?," in hypertext, 2012.

[47] Jack Edmonds. Optimum branchings. journal of research of the national bureau of standards - b. mathematics and mathematical physics vol. 71 b, no. 4, october- december 1967.

[48] L. Fratta P. M. Camerini and F. Maffioli. The k best spanning arborescences of a network.

[49] Sito web in italiano dedicato al linguaggio di programmazione python. http://www.python.it.

[50] Sito web ufficiale dedicato al linguaggio di programmazione python. https://www.python.org.

[51] Sito web dedicato alla libreria networkx. http://networkx.readthedocs.io/en/networkx-1.11.

[52] Implementazione degli algoritmi proposti in questo lavoro disponibile al seguente repository github.https://github.com/Loveuse/master_degree_thesis.

[53] *psutil* è una libreria cross-platform per ottenere informazioni sui processi in esecuzione ed utilizzo di sistema in python. https://pypi.python.org/pypi/psutil.

[54] *multiprocessing* è un package di python che offre concorrenza sia locale che remota, che supera l'effetto del *Global Interpreter Lock* di python

utilizzando sottoprocessi, invece che thread. https://docs.python.org/2/library/multiprocessing.html.