POLITECNICO DI MILANO
Facoltà di Ingegneria dell'Informazione
Corso di Laurea Magistrale in Ingegneria Informatica

Sistema di Raccomandazione di Contatti Basato su Analisi Topologiche di una Rete Sociale

Relatore: Prof. Marco COLOMBETTI
Correlatori: Ing. David LANIADO - Ing. Riccardo TASSO

Tesi di Laurea di:
Michele MONTI - matricola 710251

Anno Accademico 2008/2009

Ai miei genitori

Sommario

Con l'ascesa graduale del *Web 2.0*, evoluzione del tradizionale *Web statico* nato con l'avvento di Internet, che comprendeva solamente pagine già definite senza interazione tra il mondo virtuale e la Persona fisica, si sono sempre più diffuse le Reti Sociali online, che permettono alle Persone di ritrovarsi grazie a legami di amicizia virtuali che spesso ricalcano quelli che ci sono realmente nella vita di tutti i giorni.
In questo lavoro di Tesi si metteranno a punto delle tecniche che possano essere in grado di modellare un sistema automatico di Raccomandazione di Contatti per un utente attivo in una Rete Sociale, che si basa su tre sostegni fondamentali.
La raccolta dei Candidati da Suggerire.
La generazione della Personal Network dell'utente su cui si vuole effettuare l'analisi.
La costruzione dei Ranking attraverso l'utilizzo di differenti metriche che sfruttano l'analisi Topologica di una rete.
Tutto ciò è stato pensato prima da un punto di vista teorico e poi implementato attraverso differenti Moduli Software che rendono il tutto automatizzato.
Come ultimo, ma non per questo meno importante aspetto si testerà il sistema di Raccomandazione su un Campione Casuale di formato da Persone reali, che sarà in grado di dare una valutazione qualitativa su ogni Contatto suggerito. Ogni risposta verrà poi approfondita sotto differenti aspetti.

Ringraziamenti

Il primissimo pensiero verso le persone che voglio ringraziare è diretto ai miei genitori, che sono stati coloro che mi hanno educato, sorretto nei momenti più difficili, ma soprattutto che non mi hanno mai fatto mancare nulla e dato la possibilità, che proprio un dettaglio non è, di arrivare fino a qui al termine degli Studi.

E come non ricordarsi della nonna Ambro, da sempre punto fisso nella mia vita, nei pomeriggi di studio fin dai primi anni di Scuola e pronta a trovare un consiglio prezioso tra una merenda e l'altra.

Ora tocca agli zii Silvano e Lucy e i pischelli Matti e Pietro, gente mica troppo a posto, che in un modo o nell'altro riescono sempre a far sorridere con la loro semplicità.

E ora si passa agli amici, come Gianlu e Mino, Erni e Sak, compagni di avventura Universitaria, fatta di risate e di tutto e di più dentro e fuori al Politecnico, e da non dimenticare, anche dentro e fuori al Mundial e Architettura.

Matty, Gio, Pello, Ama e Manzo sempre presenti nelle infinite serate invernali, e non solo, in cui le partite a *PES* sono sempre state e sempre saranno vere sfide senza esclusione di colpi.

Come dimenticarsi di Getta e dei suoi molteplici storpianti soprannomi, del-

le sue lamentele su mie presunte sparizioni, delle sue molteplici espressioni e passioni che fin dalla Terza Superiore non fanno che trasmettere sempre e comunque un vortice di simpatia.
E le donzelle? Beh ci sono sicuramente anche loro, Cla, Dany, Vivi, Lucy e Anna che con un tocco di classe, ovviamente non troppa :) sono e sempre saranno le mie preferite.
E Vale, che da poco è piombata nella mia vita, ma che sembra che la conosca da sempre. Riesce sempre a trasmettermi tanta felicità e tenerezza. Di più non dico, lo faccio di persona!
David e Ricky, i due pilastri indispensabili in questo lavoro di Tesi, che son riusciti a trasmettermi una nuova e vera passione, con la loro elevata competenza e preparazione.
Il Professor Colombetti che mi ha permesso di provare una nuova e stimolante esperienza, con una Tesi veramente interessante e su misura per me.

Indice

Capitolo 1

Introduzione

Negli ultimi anni si sta assistendo ad una sempre più elevata e districata evoluzione del cosiddetto *Web 2.0*, che comprende l'insieme di tutte quelle applicazioni online che possiedono uno spiccato livello di interazione tra il mondo virtuale e le persone. Si è passati quindi dalla vecchia generazione del *Web statico*, in cui l'utente era solamente uno "spettatore" ed un utilizzatore, a una in cui sono gli utenti stessi che possono contribuire con propri contenuti multimediali.

Le Reti Sociali diffuse su Internet sono un classico esempio dell'evoluzione al Web 2.0: le persone tendono a riunirsi in queste attraverso *legami* che spaziano da una relazione di amicizia, a una di tipo lavorativo o familiare oppure anche una conoscenza solamente limitata al virtuale.

1.1 Obiettivi

Se si fa parte di una Rete Sociale è perché si vuole restare in contatto con i propri amici. Proprio per questo, spesso non è possibile spostare completamente e immediatamente la propria serie di amicizie in una rete, ma è necessario avere un aiuto che possa contribuire, per chi lo desidera, anche a trovare nuove relazioni.
Lo scopo di questo lavoro di Tesi è quello di mettere a punto delle tecniche che siano in grado di dare questo appoggio, modellando un sistema automatico di Raccomandazione di Contatti per un utente attivo in una Rete Sociale.
Mettere in pratica manualmente un lavoro del genere è sistematicamente impossibile, a causa dell'elevato numero di Persone che entrano in gioco: per questo motivo è necessario per ogni utente analizzato capire quali dati siano indispensabili e in che modo riuscire a reperirli, col fine di poter avere una completa panoramica del singolo utente. Una volta che si è in possesso delle corrette, e concrete, informazioni su ogni Persona, si passerà all'analisi dettagliata delle sue diverse caratteristiche, che non sono altro che il set dei legami descritti prima.
Con l'analisi dei risultati ottenuti, si potranno trarre delle conclusioni circa il funzionamento delle tecniche pensate e sui possibili fattori, sia positivi sia negativi, che singolarmente potranno dare il successo o l'insuccesso sul set di Contatti suggeriti.

1.2 Il Lavoro messo in pratica

I Sistemi di Raccomandazione sono molto utilizzati nella realtà attuale di Internet e nello stesso modo lo sono anche studiati. Bisogna però sottolineare che sia nell'utilizzo sia come oggetto di analisi scientifica, si vedono prevalere ampiamente le versioni che cercano di suggerire contenuti, come ad esempio la raccolta di dati e informazioni che permettono di consigliare l'acquisto online di un libro o la scelta di una meta vacanziera.
I sistemi che fanno del loro approccio la Raccomandazione di nuovi Contatti sono poco diffusi, tanto più se orientati alle caratteristiche topologiche di una rete. Questo lavoro di ricerca infatti è stato molto pionieristico per quanto riguarda la creazione delle metriche da implementare.
Senza però una struttura su cui operare, le metriche chiaramente non sono fruibili: si studierà quindi il modo ottimale per modellare la "base", su cui poi ognuna di esse andrà a lavorare con l'aiuto di potenti algoritmi, formata dal set di amicizie che già possiede una singola persona. Ci si baserà quindi sia sulle amicizie prese in modo diretto ma anche sulle interazioni che ognuno degli amici ha con gli altri.
L'unica via per verificare le caratteristiche peculiari delle metriche sarà quella di dare modo a Persone reali di metterle alla prova: è il criterio migliore che possa rendere ottimalmente l'idea sulla loro consistenza e efficienza.

1.3 Struttura della Tesi

Nel Capitolo 2 si approfondirà lo stato dell'Arte introducendo i concetti di Rete Sociale e di tutti gli studi che si possono effettuare, in modo partico-

lare sulla *Personal Network*, elemento che avrà un grosso peso in tutto il lavoro di ricerca. Questo perché le analisi Topologiche sono proprio basate su di essa, cioè quella rete che viene a crearsi considerando tutti i possibili legami di amicizia di coloro che la compongono. Verranno poi approfonditi gli Algoritmi di *Clustering*, il cui codice può essere utilizzato liberamente anche per scopi personali. Il passo successivo è quello in cui si definiscono i principali algoritmi di Raccomandazione dapprima in via generale e poi nelle Reti Sociali. In ultimo si prende in considerazione la Rete Sociale Web di *Facebook* e i suoi aspetti fondamentali.

Nel Capitolo 3 si illustrerà come si andrà ad analizzare la Personal Network da un punto di vista teorico. Si passeranno in rassegna così le cinque metriche pensate, entrando nei dettagli di ognuna.

Nel Capitolo 4 si descriveranno invece le scelte fatte nel momento dell'implementazione dei Moduli Software che andranno, prima di tutto, a reperire i candidati da suggerire e a costruire la Personal Network e poi a rendere concrete le nozioni teoriche affrontate nel capitolo precedente.

Nel Capitolo 5 si mostreranno e commenteranno i risultati dei confronti effettuati statisticamente tra le metriche e successivamente della valutazione, effettuata da ogni elemento del Campione Casuale che ha svolto il Test, mediante l'invio di un Questionario.

Infine nel Capitolo 6 si tratteranno le conclusioni del lavoro svolto e si illustreranno dei possibili sviluppi futuri alla luce dei risultati ottenuti.

Capitolo 2

Stato dell'Arte

In questo capitolo si passeranno in rassegna gli aspetti più importanti che hanno consentito di approfondire la conoscenza di base e dare la possibilità di effettuare la ricerca. Nelle varie sezioni si andranno ad approfondire le tematiche che riguardano gli studi di *Social Network*, e in particolare su *Personal Network e Clustering* per poi addentrarsi nella tematica dei *Sistemi di Raccomandazione* e infine dare un accenno sul mondo di *Facebook*.

2.1 Studi di Social Network

Una *Social Network*, come dice *Wasserman* in [22], consiste in uno o più set finiti di attori e una o più relazioni definite su di essi. Si definiscono immediatamente i concetti chiave: con *attore* si intende l'insieme che comprende "individui, aziende o unità sociali collettive", come ad esempio persone di un gruppo o di un dipartimento universitario, mentre con *relazione* si intende il legame che c'è tra due o più attori. Ci sono vari tipi di legami sociali:

- Di amicizia
- Transazioni aziendali
- Partecipazione a un evento o l'iscrizione a un club
- Partecipazione a newsgroup
- Fisici come strade, fiumi o ponti

Nella prossima sotto-sezione verrà introdotta la definizione di Personal Network, che è un sottinsieme finito all'interno di una Social Network stessa.

2.1.1 Personal Network

Il concetto di Personal Network definisce quella parte di una Rete Sociale che si incentra su di un singolo utente (utente centrale), definendo per esso tutte e sole le possibili connessioni del primo ordine. L'utente centrale sarà collegato ai propri amici, che vanno così a definire la lista delle amicizie.

Come dicono *Wasserman* e *Faust* in [22], possono essere effettuate delle misure su tali legami, che essi siano tra il nodo centrale e la lista completa oppure atti a collegare più amici all'interno della lista.
Ogni persona può essere in collegamento a sua volta con altri nodi amici (e così via), che però non verranno considerati come appartenenti alla Personal Network iniziale. Ogni utente quindi potrà condividere i contenuti che i nodi della propria Personal Network rendono pubblici e accessibili a tutti, secondo le proprie impostazioni sulla Privacy e che, nei casi di nodi attivi su reti aziendali, possono accrescere le proprie informazioni.

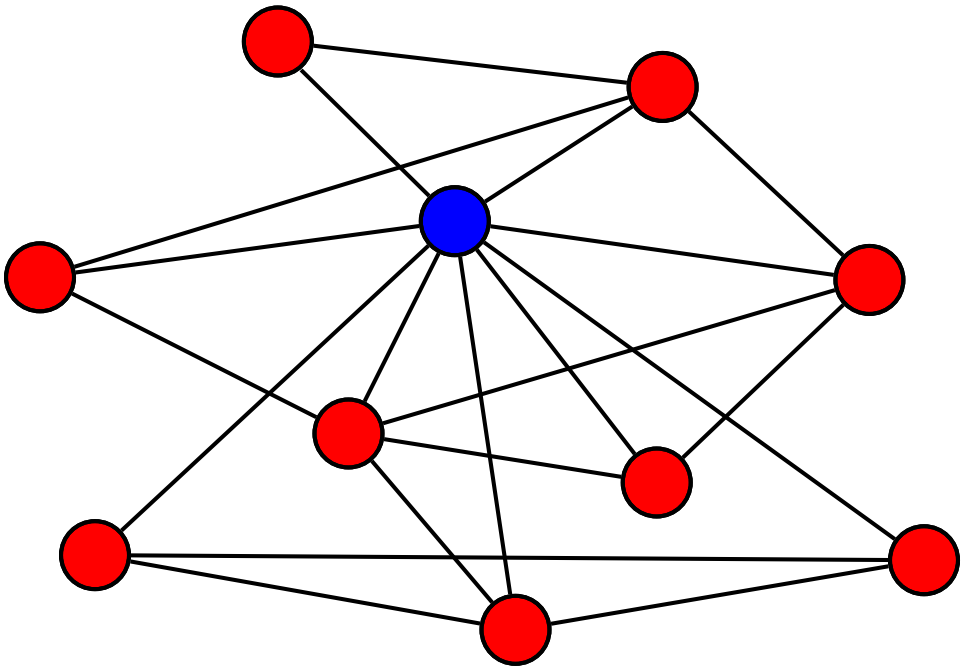

Figura 2.1: Esempio di Personal Network dove il nodo in blu è l'utente centrale e quelli rossi rappresentano le amicizie

Struttura e Analisi Le Personal Network hanno una struttura vincolata e semplice che permette un potente mezzo nel momento dell'analisi, anche se come rovescio della medaglia hanno una carenza qualitativa di informazioni che reti più grandi avrebbero tra i propri punti di forza. Le tecniche di

analisi si incentrano sulla *densità*, sulla *connettività*, sugli *attributi* di nodi e connessioni, oppure su un mix delle tre caratteristiche [5]. *Freeman*, già nel 1978 in [7, 8] introdusse il concetto di *centralità*, quantità puramente matematica legata alla densità, che si rivelò poi fondamentale per determinare i nodi con più importanza all'interno della rete. *Ibarra* invece in [9] definisce il concetto di centralità come quella funzione che assegna dei valori (compresi nel range [0,1]) in base alla vicinanza di un nodo a altri nodi con una grande centralità già accertata.

Vertex Betweenness La *Betweenness* è quella caratteristica di *centralità* che ogni nodo possiede. Essa è quantitativamente calcolata come il numero di percorsi passanti tra una coppia di nodi che transitano attraverso il nodo a cui si è interessati[5].
Se si volesse invece dare una definizione più qualitativa, essa la si può definire come il livello di influenza che un tal nodo ha nell'essere un punto di passaggio verso il resto della rete, come illustra *Newman* in [14].

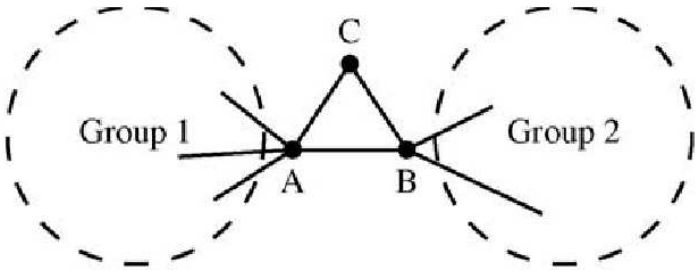

Figura 2.2: In questo esempio A e B hanno un alto valore di betweenness perché sono di passaggio tra i gruppi 1 e 2, a differenza di C che è solamente collegato con A e B.

La formula più ricorrente per il calcolo di tale caratteristica è la seguente:

$$b_i = \frac{\sum_{s<t} \frac{g_i^{st}}{n_{st}}}{0.5n(n-1)}$$

dove:

- g_i^{st} è il numero di percorsi tra il nodo s e il nodo t che transitano da i
- n_{st} è il numero di percorsi totali tra s e t
- n il numero di nodi totali della rete

2.1.2 Algoritmi di Clustering

Gli *Algoritmi di Clustering* sono utilizzati per determinare un raggruppamento logico dei nodi all'interno di una rete, in modo tale che essi possano essere partizionati in differenti comunità, ognuna riconosciuta secondo le caratteristiche proprie dell'algoritmo stesso. Non esistono delle definizioni formali di *comunità* (oppure *cluster* oppure *sottogruppi coesi*), ma si tende sempre a descriverla come quel gruppo di persone che condividono le stesse amicizie, quindi che presenta legami forti [6]. Questi legami possono essere identificati attraverso l'utilizzo di un'analisi sulla struttura della rete con il calcolo della *modularità* (v. prossimo paragrafo). Ci si concentra quindi sugli archi, che devono avere caratteristiche simili a quelli che collegano gli altri nodi.

Modularità La *modularità* è quella caratteristica che più garantisce, nell'analisi di rete, l'espressione del grado di coesione di ogni comunità: essa è la quantità da massimizzare per ottenere la miglior struttura possibile. È particolarmente utile come parametro perché il processo di massimizzazione è molto veloce da ottenere anche con grandi e complesse reti. *Newman* ha introdotto il concetto di *modularità* in [15, 13] come quel valore dato dalla

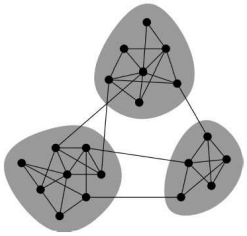

Figura 2.3: Esempio pratico di suddivisione in cluster

differenza tra il numero di archi che collegano nodi dello stesso tipo (quindi nello stesso cluster) e il numero previsto da una rete differente, con le stesse comunità ma connessioni casuali tra i nodi. Tale valore potrà quindi essere positivo e compreso in [0,1], se la rete presenta suddivisioni: il range di valori perché una rete sia suddivisa qualitativamente bene si concentra tra 0.3 e 0.7. Oltre quella soglia, la rete avrebbe legami troppo forti e quindi il clustering potrebbe risultare non sufficientemente corretto. Se il valore risulta negativo, la rete non presenta alcuna suddivisione in cluster.

La formula della modularità è la seguente:

$$Q = \frac{1}{4m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j$$

con:

- m numero totale di archi della rete
- A_{ij} matrice di adiacenza degli archi, $A_{12} = 1$ se c'è un arco tra 1 e 2
- k_x grado del vertice x
- s_p appartenenza o meno ad un gruppo p

In questo lavoro di Tesi ne sono stati analizzati quattro differenti, che verranno descritti nelle prossime sotto-sezioni.

Walktrap

L'idea principale di questo algoritmo è quella che, se esistono dei cammini casuali tra un nodo e un altro, essi devono tendere a propagarsi all'interno di determinate comunità connesse. La *distanza* di ogni cammino può essere calcolata efficacemente e passata come parametro all'interno di un algoritmo gerarchico, che determina poi la suddivisione in comunità attraverso la generazione di un dendrogramma.

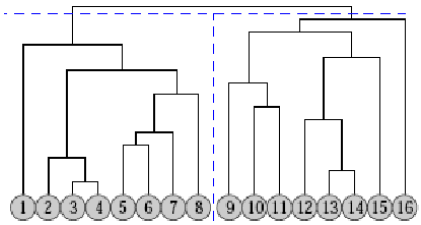

Figura 2.4: Esempio di dendrogramma

Dopo aver effettuato una misura della somiglianza tra vertici, l'algoritmo in [17] raggruppa iterativamente i vertici in ogni cluster.

Elementi dell'algoritmo:

$A_{i,j} = 1$ (se i nodi i e j sono collegati)

$d(i) = \sum_j A_{i,j}$ (numero di vicini di i incluso se stesso)

$P_{i,j} = \frac{A_{i,j}}{d(i)}$ (probabilità che ci sia un arco tra i e j, dove P è la *matrice di transizione*)

$P^t_{i,j}$ (probabilità di andare da i a j con cammino random di lunghezza t)

$P^t_{C,j} = \frac{1}{C} \sum_i P^t_{i,j}$ (con i∈C, probabilità di andare dalla comunità C al vertice j in t passi)

Il Walktrap comincia prendendo in esame una partizione $P_1 = \{\{v\} \in V\}$, di n comunità composte da un singolo nodo, tra cui vengono calcolate per ognuna la distanza verso i nodi adiacenti. La partizione poi evolve, a ogni k, seguendo queste direttive:

- scegliere due comunità C_1 e C_2 in P_k utilizzando i valori delle distanze;
- fare il merge delle due comunità $C_3 = C_1 \cap C_2$, creando la nuova partizione: $P_{k+1} = (P_k \setminus \{C_1, C_2\}) \cap \{C_3\}$;
- aggiornare la distanza tra le comunità. Dopo $n - 1$ passi, l'algoritmo finisce e si ottiene $P_n = \{V\}$.

Ogni step definisce così un partizionamento del grafo in comunità e viene realizzato così il *dendrogramma*.
Gli esperimenti condotti su questo algoritmo hanno mostrato che il metodo viene supportato da ottimi risultati sotto differenti punti di vista. Non ci sono stati problemi con elevate dimensioni della rete, con densità differenti e sul numero dei cluster ottenuti. Da non sottovalutare è il risultato che si può avere nel rapporto tra il tempo necessario per calcolare le comunità e la qualità della loro suddivisione. C'è uno svantaggio, posizionato nel quantitativo di memoria che l'algoritmo utilizza, elevato per analisi di grosse reti. La complessità del Walktrap si attesta per l'appunto in un tempo $O(mn^2)$ e spazio $O(n^2)$ nel peggior caso, e in un tempo $O(n^2 \log n)$ e spazio $O(n^2)$ nei casi di uso più diffusi (con n numero di nodi e m numero di archi).

Edge Betweenness

Newman e *Girvan* in [13] hanno messo subito in risalto la differenza tra gli algoritmi di clustering *agglomerativi*, o *gerarchici*, e quelli di tipo *divisionale*. Quelli agglomerativi (v. Walktrap) diverse volte falliscono nel momento di trovare le corrette comunità, perché tendono a priori a ignorare i nodi più periferici. Negli algoritmi divisionali invece la scelta è quella di iniziare da coppie di nodi con caratteristiche simili eliminando l'arco tra di essi. Ripetendo in modo continuo questa operazione, si riesce a dividere la rete in piccole comunità, ottenendo comunque un risultato concreto in un qualsiasi passo a scelta.
L'approccio dell'*Edge Betweenness* lavora però in modo lievemente differente e consiste nel determinare gli archi nella rete che stanno di più "tra i nodi", in modo che siano responsabili a loro volta del collegamento di molte coppie di altri nodi. L'algoritmo si basa sulla *betweenness*, cioè quella misura che dà più peso agli archi che sono da collegamento tra i diversi cluster a scapito di quelli tra nodi interni agli stessi cluster.

Ecco di seguito i passi in via generale dell'algoritmo:

1. Calcola il valore di beetwenness per ogni arco della rete
2. Trova l'arco con il valore più alto e lo rimuove dalla rete (se più di uno con lo stesso valore, se ne rimuove uno a caso)
3. Ricalcola il valore di betweenness per gli archi restanti
4. Ripete dal punto 2 ciclicamente.

Lo studio ha portato alla conclusione che il passo 3. (ricalcolo valori) è il punto cruciale dell'algoritmo, in quanto garantisce una potenza e una

precisione nella determinazione delle comunità molto elevata se messo a confronto con una versione dell'algoritmo che non presenta tale passo.

C'è come in tutte le cose un punto debole, che nell'*Edge Betweenness* è quantificato nell'ingente lavoro computazionale che richiede. La versione più veloce opera in un tempo dell'ordine di $O(m^2n)$ con reti regolari e $O(n^3)$ con reti sparse (tenendo in considerazione m numero di archi e n quello di nodi). È meglio evitare reti con oltre 10000 nodi, perché sarebbe troppo ingente il tempo computazionale. Un metodo per ovviare tale inconveniente potrebbe essere quello di spostare il calcolo su di una macchina parallelizzata, frazionando così il carico su più unità di elaborazione. *Tyler* in [10] spiega un suo metodo che consiste nel prendere un set casuale di nodi ed eseguire il calcolo su di esso. Tale set avrà una grandezza basata sulla rete e campionata finché non si trova un arco con il valore della betweenness al di sopra di una soglia minima.

Fastgreedy

Per l'algoritmo *Fastgreedy* la *modularità* ha un compito importante perché per alti valori, la divisione in comunità è buona. Essendo un algoritmo di tipo *greedy*, quindi "che ottiene la soluzione ottima a livello globale prendendo la migliore possibile a ogni passo locale", come è illustrato in [4], si parte dalla suddivisione della rete in comunità composte da un solo nodo, per poi unire di volta in volta le due che producono, accoppiate, il più alto valore di modularità. Al massimo in $n-1$ passi, con n numero di nodi, si ottiene la soluzione con singolo cluster e l'intero processo viene rappresentato, come per gli altri algoritmi, attraverso la generazione di un *dendrogramma*.

Elementi dell'algoritmo:

$\Delta A_{v,w}$ (matrice di adiacenza)

ΔQ (matrice che contiene i valori $\Delta Q_{i,j}$)

$\Delta Q_{i,j}$ (variazione di modularità per per ogni coppia di comunità i,j)

a_i (vettore ordinario)

H (lista che contiene il massimo valore per ogni riga di ΔQ)

Ecco i passi dell'algoritmo nel dettaglio:

1. Calcola $\Delta Q_{i,j}$ e a_i popolando il gruppo con gli elementi con gli elementi di ogni riga della matrice ΔQ.

2. Seleziona il più grande $\Delta Q_{i,j}$ da H, crea la corrispondente comunità e aggiorna la matrice ΔQ, il gruppo H a_i incrementando Q di $\Delta Q_{i,j}$.

3. Ripete il passo 2. fino a $n - 1$ cicli.

Il *Fastgreedy* opera in tempo $O(md \log n)$ con reti sparse, $O(m \log^2 n)$ per reti bilanciate e infine $O(n \log^2 n)$ per reti bilanciate e sparse (con m numero di archi e d numero di traiettorie sparse). E' considerabilmente veloce come computazione e può essere utilizzato in reti che, per la loro grandezza (anche milioni di nodi), non potrebbero essere analizzate da altri algoritmi troppo pesanti a livello computazionale. Oltre all'efficienza, quindi, si arriva *efficacemente* però a determinare in modo chiaro la suddivisione in cluster della rete, cosa fondamentale.

Spin Glass

Come affermano *Reichardt* e *Bornholdt* in [18], sono nati nell'ultimo periodo dei metodi di clustering che si ispirano a meccanismi statistici o che hanno analogie con modelli fisici. L'algoritmo *Spin Glass* è basato sul significato statistico della ricerca delle comunità e di come essa sia in relazione al partizionamento di grafi: nell'implementazione è utilizzato come parametro fondamentale l'energia dei *Sistemi di Spin* [19], in cui ogni stato del sistema corrisponderà a un determinato raggruppamento della rete.

Elementi dell'algoritmo:
$H(\{\sigma\})$ (*Hamiltoniano*: valore derivato dalle componenti positive o negative date da ogni arco della rete, che esso sia interno o esterno alle varie comunità o che colleghi differenti comunità)
$Q = \sum_s e_{s,s} - a_s^2$ con $a_s^2 = \sum_r e_{r,s}$ (*Modularità*: $e_{r,s}$ è la frazione di link che esiste tra i nodi nei gruppi r e s, in altre parole la probabilità che un arco colleghi un nodo di r e uno di s; mentre a_s è la probabilità che un link finisca nel gruppo s)

Partendo da un *ansatz*[1] si trova lo Spin Glass che poi porta a determinare il valore di modularità ottimale. In base a tale valore, l'algoritmo torva così le proprietà strutturali che poi avranno i cluster. In base alla variazione dei parametri iniziali, si determinano le comunità osservando così le caratteristiche comuni dei nodi, che dunque verranno inclusi in un particolare cluster piuttosto che in un altro.

[1] E' il punto di partenza che aiuta a determinare i parametri fisici in gioco

2.2 Algoritmi di Raccomandazione

I *Sistemi di Raccomandazione* sono entrati a far parte di un'importante area di ricerca, che ha creato negli anni una moltitudine di studi, effettuati sia da aziende sia a da Dipartimenti Universitari a partire dalla seconda metà degli anni '90. La generazione attuale è impegnata nel riuscire a implementare metodi per poter suggerire al meglio nei diversi aspetti della vita comune, come a esempio le vacanze o gli investimenti bancari mirati e aggiungendosi a quelli già più diffusi, che inducono l'utente ad acquistare un bene piuttosto che un altro.

Il problema principale da risolvere per un Algoritmo di Raccomandazione è quello di determinare il *rating* da assegnare a ogni tipologia di oggetto che l'utente non ha già incontrato o di cui non è ancora a conoscenza: una volta trovato il valore, il suggerimento va esplicitato attraverso più oggetti simili della stessa categoria, ordinati in base al rating più alto. Ci sono differenti metodi che eslpicano diverse tipologie di raccomandazione: *Adomavicius* in [1] sottolinea l'importanza di quelli *orientati ai contenuti*, che mettono in risalto oggetti simili a quelli in passato acquistati dall'utente, quelli *collaborativi*, che suggeriscono oggetti che rispondono ai gusti e alle preferenze passate dell'utente, e quelli *ibridi*, con caratteristiche miste dei due precedenti.

2.2.1 Tipologie

Orientati ai contenuti In questa metodologia il rating si attesta attorno al valore dell'*utilità* che per l'utente c e la varietà degli item $s_i \in S$ ha questa funzione: $u(c, s_i)$. Tale valore dell'utilità è calcolato secondo:

$$u(c, s) = Score(ContentBasedProfile(c), Content(s))$$

dove $ContentBasedProfile$ può essere definito come un vettore che contiene i pesi $w_{c,k}$ relativo agli interessi k dell'utente c, mentre $Content$ definisce il set di attributi di un oggetto da suggerire.
Ci sono però altri approcci che non si basano solo su valori di euristiche ma solo su modelli che attingono i dati usando tecniche statistiche meccanizzate, che riescono a identificare i contenuti "rilevanti" da quelli "non rilevanti", come il *classificatore Bayesiano* [20].
I limiti di questa tipologia di suggeritore si instaurano nel momento dell'estrazione dei dati necessari: siccome lavorano utilizzando solo testo, nel momento in cui i contenuti hanno un formato multimediale (immagini, audio, video) essi risultano inefficaci. Un altro difetto è quello che se due differenti oggetti sono descritti dalle stesse caratteristiche, possono risultare identici all'algoritmo, nonostante invece siano effettivamente oggetti diversi.

Collaborativi Questa seconda metodologia vede il rating sempre calcolato come valore dell'*utilità*, ma in modo duale rispetto al precedente. La varietà non è sugli item che l'utente preferisce, ma sui i gusti e le preferenze, nel tempo, degli utenti simili $c_j \in C$ verso ogni item s e ha questa funzione: $u(c_j, s)$.
Il *Collaborative Filtering* è il sistema di raccomandazione collaborativo più usato nel Web. Esso agisce utilizzando tecniche statistiche, come la *Correlazione di Pearson* [16] , applicate sul vicinato (*neighborhood*) dell'utente

analizzato, per avere informazioni sulle preferenze dei contatti più stretti. Come dice *Badrul* in [2] i vicini possiedono informazioni che "concorrono a completare la profilatura dell'utente dell'oggetto da suggerire".
Ecco alcuni parametri:

- *Previsione* di quanto un utente gradisce un prodotto P:

  $C_{P_{pred}} = \bar{C} + \frac{\sum_J (J_p - \bar{J}) r_{C,J}}{\sum_J |r_{C,J}|}$, dove C è l'utente, J è il vicino, $\bar{C}$ e $\bar{J}$ ne sono i valori medi e r_{CJ} è la correlazione tra l'utente e il vicino

- *Raccomandazione* di una lista di prodotti per l'utente (chiamata *top-N*)

Oltre ad avere grosso potenziale, questa categoria di algoritmi possiede anche delle limitazioni, che sono identificate nella *sparsità*, difetto dovuto al grosso volume di informazioni che inevitabilmente porta a considerare parte degli oggetti da suggerire a scapito di altri, nella *scalabilità*, che interviene nel momento in cui i dati crescono a dismisura e l'algoritmo non riesce a gestire il relativo accrescimento computazionale, e nella *sinonimia*, caso in cui più oggetti abbiano nomi simili che non possono essere riconosciuti univocamente dall'algoritmo.

Ibridi Ci sono vie differenti per creare sistemi di raccomandazioni misti tra *Orientati al contenuto* e *collaborativi* e possono essere classificati nelle seguenti tre varianti:

1. Implementazione dei due differenti algoritmi combinandone infine le previsioni

2. Incorporare le caratteristiche di uno nell'approcio dell'altro

3. Costruire un modello generale unificando sia le caratteristiche di uno sia le caratteristiche dell'altro

Vediamo ora in un riassunto tabellare, presente anche in [1], le suddivisioni dei diversi approcci e a loro volta le principali differenze tra tecniche *a euristiche* e *a modelli*.

Approccio	Tecniche a euristiche	Tecniche a modelli
Orientati ai contenuti	1. TD-IDF 2. Clustering	1. Bayesiano 2. Clustering 3. Alberi Decisionali 4. Reti Neurali Artificiali
Collaborativi	1. Vicinato 2. Clustering 3. Teoria dei grafi	1. Reti Bayesiane 2. Clustering 3. Reti Neurali 4. Regressione Lineare 5. Probabilistici
Ibridi	1. Combinazione lineare 2. Schemi di voto 3. Composizione da euristiche diversi	1. Modello unificato 2. Composizione da modelli diversi

Tabella 2.1: Sistemi di Raccomandazione

2.2.2 Nei Social Network

Nei *Web Social Network* è disponibile la possibilità di aggiungere alla propria "lista amici" delle nuove connessioni, con lo scopo di ingrandire e articolare la propria *Personal Network*. Principalmente, nella Personal Network, ogni utente tende a aggiungere prima di tutto le proprie amicizie pregresse, ovvero quelle che sono già consolidate in un momento precedente all'accesso alla rete sociale (le cosiddette amicizie *offline*). Come dicono *Chen et al.* in [3], le nuove conoscenze, e quindi le nuove connessioni, possono essere suggerite all'utente: non è cosa facile come suggerire un libro o un film, perché a differenza di questi ultimi, bisogna mettere in conto tutte le implicazioni che possono sorgere nel momento in cui magari una persona non riconosce il possibile amico o il suggerimento porta a una persona sconosciuta o quasi. In [3] è stato condotto uno studio su *Beehive*[2], una Social Network creata da *IBM* e lanciata nel luglio 2007. Essa è concettualmente molto simile a *Facebook* e quindi può essere un buon livello di paragone. La sostanziale differenza sta nella tipologia di connessione sulle amicizie: in Beehive possono essere asimmetriche, cosa che in Facebook non può avvenire, quindi potenzialmente un utente può essere può essere amico di tutti, senza che questi lo vengano neanche a sapere. Lo studio è stato poi completato andando ad analizzare quattro differenti algoritmi di *Friend Recommendation*.

Content Matching Questo metodo è basato sulla condivisione di contenuti: in altre parole se due persone non connesse pubblicano contenuti simili, possono essere messe in condizioni di mettersi in collegamento l'una con l'altra. L'algoritmo crea una sorta di "recipiente" con una serie di parole prese tra i contenuti (status, link, foto) e le informazioni personali (luogo dove la persona analizzata vive, studia o lavora) e lo mette in relazione,

[2]http://www.ibm.com

attraverso la tecnica TD-IDF (v. Tabella 2.1), con gli altri utenti a cui non è ancora connesso:

$$v_u(w_i) = TF_u(w_i)\, IDF_u(w_i) \text{ con:}$$
$$w_x \text{ parola usata}$$
$$TF_u(w_i) = \frac{\#w_i}{\#w_u}$$
$$IDF_u(w_i) = \log \frac{\#utenti_{tot}}{\#utenti_{w_i}}$$

Il confronto tra due utenti a e b è fatto tra i vettori V_a e V_b, così definiti:

$$V_x = [v_x(w_1), ..., v_x(w_m)]$$

Se i vettori V sono simili, significherà che il suggerimento di amicizia può essere preso in considerazione.

Content-plus-Link Il lavoro di questo algoritmo è sulla falsa riga di quello già descritto nel paragrafo precedente ma ha un riguardo in più circa i link tra due differenti utenti. Per *link*, Chen et al. [3] intendono una sequenza di utenti. Per essere in sequenza, due utenti a e b devono:

1. a essere collegato a b
2. a commentare qualcosa di b
3. b connettersi ad a

Un esempio pratico di sequenza può essere: "Alice ha commentato qualcosa di Bob, che è considerato un amico di Charles."

Friend-of-Friend Questa tipologia di algoritmo invece è basato sul funzionamento del "Persone che potresti conoscere" direttamente implementato e in uso su Facebook[3].
Il *predicato* $F(a,b)$ è vero se c'è una connessione di amicizia tra l'utente a e quello b, mentre si possono definire le seguenti leggi:

- $RC(u) = \{utente\ c\ t.c.\ se\ \exists\ a\ F(u,a) \wedge\ F(a,c)\}$
- Per ogni c candidata $\in RC(u)$, il suo set di amici mutui è:

  $MF(u,c) = \{utente\ a,\ se\ F(u,a) \wedge\ F(a,c)\}$ dove l'utente a sta a indicare che è l'amico che fa da collegamento tra l'utente u e l'utente c.

La dimensione di $MF(u,c)$ è lo score del possibile suggerimento all'utente c, considerando l'insieme delle candidate $RC(u)$.

SONAR L'algoritmo è basato sul sistema *SONAR*, che aggrega differenti informazioni sociali attinte da varie risorse di *IBM*. Lo score per una possibile nuova connessione cade nel range [0,1], dove se è 0 la relazione non c'è, mentre se è 1 la relazione è importante e da prendere in considerazione. Esso è calcolato basandosi sulla bontà e la frequenza delle interazioni su di una informazione.

[3]Così dice l'articolo ma non possiamo esserne certi perché Facebook non è OPEN SOURCE

Analisi dei risultati

L'indagine è stata sottoposta da *Chen* a 500 utenti e in 258 hanno testato i quattro algoritmi. Il 95% ha ritenuto che un suggeritore di amicizia sia una cosa veramente furba e utile all'interno di una Social Network.
Per ogni raccomandazione, ogni utente è stato in grado di individuare se la persona suggerita fosse conosciuta o meno oppure se la raccomandazione stessa fosse buona.
Dai risultati di *Content* è uscito che il 77.6% dei suggerimenti portavano a persone sconosciute, di cui solo il 30.1% "una buona raccomandazione". Per quanto riguarda *CplusL* la cifra degli sconosciuti si abbassa al 64.8%, mentre tra i conosciuti (35.2%), quasi tutti sono considerati "un buon suggerimento". Per quanto riguarda invece *FofF* la quota dei suggerimenti di conosciuti si assesta al 60%, di cui praticamente tutti (55%) sono una "buona raccomandazione". Passando in rassegna anche *SONAR*, quasi tutti i suggerimenti (86%) sono di gente conosciuta, di cui l'81% è considerato come una "buona raccomandazione".
Da ciò si può capire come i suggeritori *Orientati ai Contenuti* siano in grado di raccomandare persone spesso sconosciute, ma comunque in altissima percentuale vengono definite come un buon suggerimento. Andando a vedere i risultati degli altri due algoritmi, più orientati alle interazioni, le persone suggerite sono molto più spesso conosciute e quasi sempre il suggerimento è considerato buono. Il che, ragionando un attimo, è normale che avvenga e che sia così.

2.3 Facebook

Facebook è un sito web di Social Network gratuito attivo dal 2004 e accessibile a chiunque.
Il nome del sito deriva dalla denominazione che hanno gli annuari distribuiti nei college statunitensi con le foto dei membri per permettere a tutti di conoscere le persone all'interno dei campus.
Inizialmente Facebook[4] è nato con lo scopo di mettere in comunicazione le diverse comunità all'interno dei college e, partendo da Harward, è arrivato fino a Yale, Stanford e si è radicato in varie aziende fino ad arrivare nel 2006 a coprire trasversalmente gran parte della popolazione, esclusa la Cina, che usa Internet attivamente. Attualmente infatti sono circa 400 milioni gli utenti registrati e è stimato che abbiano portato il sito a un valore che si aggira su dieci miliardi di dollari e proprio nel 2009 ha ottenuto per la prima volta l'attivo in bilancio[5].

2.3.1 Caratteristiche e struttura

Ogni utente ha la possibilità di completare il proprio profilo scegliendo quali sezioni modificare tra le tipologie disponibili, che esse siano di base (data e città di nascita, orientamento politico e religioso), oppure orientate agli interessi personali. Si possono anche inserire i vari canali di contatto (e-mail

[4]http://www.facebook.com
[5]http://www.wikipedia.org/wiki/Facebook

Figura 2.5: Homepage di Facebook

e contatti di messaggistica instantanea) e le informazioni circa gli studi e le esperienze lavorative effettuati o in corso.
In aggiunta alle informazioni, è presente anche una Bacheca su cui si può aggiornare lo stato personale, scambiare messaggi pubblici, pubblicare gli album delle proprie foto, i propri video, condividere link e invitare gli amici a partecipare a un evento.
Esiste anche un sistema di messaggistica privata che permette di scambiarsi informazioni personali senza che tutti gli utenti possano vederli.

Il meccanismo per aumentare la propria lista di amici è di tipo *simmetrico*: per aggiungere un nuovo contatto, lo si può fare inviando una richiesta di amicizia che verrà poi confermata o ignorata in un momento successivo dall'altro utente. Ovviamente chiunque può inviarci una richiesta e saremo

noi a confermarla o a ignorarla. Sono già presenti due applicazioni che permettono di trovare più facilmente nuovi collegamenti attraverso un duplice suggeritore, perché lo scopo di Facebook è quello di "aiutarti a trovare e mantenere i contatti con le persone della tua vita". Si possono trovare nuovi amici utilizzando il tool "Persone che potresti conoscere", direttamente implementato da Facebook e che sfrutta gli amici in comune per suggerire nuove amicizie: il suo algoritmo non è Open Source[6] e quindi non reperibile; esso utilizza come parametri anche il tempo, inteso sia da quanto un Utente abbia attivato un profilo, ma anche relativo a quanto sia frequente l'ampliamento della sua lista amici. Oltre a tutto ciò si può anche effettuare una ricerca sfruttando gli indirizzi e-mail della propria rubrica o dei contatti di messaggistica istantanea. Ogni utente può essere in collegamento con un numero preciso di amici e ha un limite massimo, che è di 4999 connessioni. Attraverso la funzionalità di "ricerca" si possono trovare anche i contenuti, come le *Pagine* e i *Gruppi*.

Cardinalità La stima della cardinalità delle proprie amicizie è individuata da Facebook mediamente nei 120 individui: bisogna tenere conto che non tutte le proprie conoscenze possono essere presenti su Facebook e che quindi mai si potrà essere in grado di costruire una rete personale completa. Per poter dare un senso numerico e poter dimensionare la rete, ci si può riferire al lavoro di *Killworth*, descritto in [11], che nel suo studio ha testato diversi individui nel riconoscere le proprie amicizie. Analizzando una lista di nomi e cognomi casuali, presa da un elenco telefonico, la dimensione media ottenuta in ogni singolo test varia dalle 300 alle 3000 persone. Secondo *Roberts* in [21] esiste una relazione tra la dimensione della propria rete e la componente emotiva: le reti con pochi nodi hanno connessioni omogenee e chiuse,

[6]Indica un software i cui autori ne permettono, anzi ne favoriscono, il libero studio e l'apporto di modifiche da parte di altri programmatori indipendenti. Questo è realizzato mediante l'applicazione di apposite licenze d'uso.

mentre le reti più grandi presentano caratteristiche socialmente deboli. Per portare a compimento questa teoria, *Lu* e lo stesso *Roberts* in [12] hanno preso in esame un campione di 30 studenti pre-universitari e caratterizzato ciascuno utilizzando un questionario che mettesse specificatamente in risalto le componenti circa la personalità di ognuno. Utilizzando metodi statistici, lo studio ha permesso di legare la dimensione della rete a fattori di tipo personali che riguardano il carattere di un individuo nell'essere propenso a connettersi con gli altri.

2.3.2 Privacy

La Privacy[7] di ogni utente di Facebook è regolata attraverso una serie di livelli personalizzati che, se necessario, inibiscono la visualizzazione di tutto il profilo o di uno o più contenuti del profilo stesso. Applicando correttamente le varie impostazioni, si può ottenere una limitazione della diffusione di dati personali. Infatti nel menù *Impostazioni sulla Privacy* si può differenziare profilo, ricerca, notizie e bacheca. E' chiaro anche che la limitazione nel diffondere dati personali può essere imposta sui contenuti del proprio profilo ma non su quello degli altri. Ogni contenuto che un utente della rete carica viene automaticamente reso di proprietà di Facebook: tale concetto però non implica la presa di responsabilità, in quanto non viene applicata nessun tipo di censura o limitazione a gruppi o pagine.

Altra caratteristica importante da sottolineare è che ogni utente può scegliere se rendere visibile la propria lista amici a tutti coloro che possiedono un account, a meno di restrizioni con cui solamente i propri amici possono

[7] http://www.facebook.com/policy.php

visualizzarla. Queste informazioni sono utilizzate da Facebook stesso per proporre le notizie che sono reputate tra le più interessanti.

Sul sito stesso di Facebook viene dichiarato che oltre alle informazioni immesse dall'utente, vengono registrati a ogni accesso l'indirizzo IP e le informazioni relative al browser. Il nome, i nomi delle reti di cui si fa parte e l'indirizzo e-mail saranno utilizzabili per comunicazioni di servizi offerti da Facebook e possono essere messe a disposizione di motori di ricerca di terzi.

Capitolo 3

Metriche per Determinare i Ranking

Facebook è una rete sociale e, in quanto tale, è possibile immaginarla come un *grafo* che la rappresenta. Un grafo è un insieme di elementi chiamati nodi, collegati tra loro da archi. Se si vuole dare una formulazione rigorosa, un grafo è la tupla $G = (V,\ E)$, con V insieme dei nodi ed E insieme degli archi. E è formato da coppie di nodi (x, y) con x e $y \in V$. Una relazione di amicizia di Facebook quindi è l'arco che connette due nodi.

Per riuscire a raggiungere l'obiettivo della ricerca, si sono scelte cinque differenti metriche basate sull'analisi *Topologica*[1] della Personal Network, col fine di determinare una serie di "Persone da Suggerire". Tali persone sono quelle che ogni metrica riconosce come le migliori e quindi con le proprie

[1] Si basa sulla proprietà delle forme del grafo, quindi studia la disposizione di vertici e archi.

valutazioni più alte. Verranno poi suggerite all'utente su cui ci si sofferma durante l'analisi (d'ora in poi *nodo centrale* o *ID*) e saranno chiamate *nodi candidati*. È importante sottolineare che sono tutti nodi esterni alla Personal Network del nodo centrale, perché sono dei nodi "Amici di Amici".

Si è partiti con l'analizzare sia la rappresentazione della Personal Network del nodo centrale (d'ora in poi *rete G1*), sia la rete estesa anche agli "Amici di Amici" (d'ora in poi *rete G2*), implementando il codice necessario in grado di stilare, per ogni singola metodologia, una classifica di possibile gradimento do ogni candidato appartenente alla rete G2. Questo dando chiaramente peso ogni volta ad aspetti differenti al momento dell'analisi.

Ora è il momento di definire i concetti che portano a capire meglio cosa si intende con i termini "rete G1" e "rete G2":

- $SonoAmici(x,\ y)$: "x e y sono amici "
- $A\,(x)$: "Amici di x"
- $B = A\,(ID) = P^{(ID)}$, "Amici di ID"
- $P^{(ID)} = \{b_1,\ b_2, ...,\ b_m\}$, lista degli m amici di ID
- $S^{ID} = \{c_1,\ c_2, ...,\ c_n\}$, lista dei nodi candidati c_n
  dove $\forall c_n$: $\exists\ b\ t.c.\ b \in B \wedge SonoAmici\,(b,\ c_n)$
- $G1 \supseteq P^{ID}$ ed è un grafo con:
  - NODI: $A\,(ID)$
  - ARCHI: (x, y) con $(x = ID,\ y \in A\,(ID)) \vee (x, y \in A\,(ID) \wedge SonoAmici\,(x,\ y))$
- $G2 \supseteq \left(P^{ID} \cup S^{ID}\right)$ ed è un grafo con:

- NODI: $A(ID) \cup \{c \;\; t.c.\; c \in A(b), b \in A(ID)\}$
- ARCHI: (x, y) con $x = ID,\; y \in A(ID) \vee x \in A(ID),\; y \in A(x)$

Ora ci soffermeremo nello spiegare dettagliatamente ogni algoritmo ideato, ognuno dei quali prende in ingresso l'insieme delle liste $A(c_i)$ e che a sua volta contengono le connessioni di ogni nodo candidato i, dove $i = 1,\ ...,\ n$. Con $A(c_i)$ si può anche dare la definizione di *neighborhood* (vicinato) del nodo c_i, in quanto ogni suo elemento $b \in b_m$ è a "distanza 1" da c_i stesso.

3.1 Amici in Comune

Questa metrica è abbastanza semplice sia da formulare e calcolare sia da comprendere intuitivamente. Essa determina il numero di amicizie in comune che il nodo analizzato condivide con ogni nodo candidato ed è stata studiata, come già visto nella sotto-sezione 2.2.2, anche nel lavoro di *Chen* in [3]. In questo modo ci si fida quindi delle amicizie del nodo centrale: tutti all'interno della Personal Network hanno lo stesso peso e si prende ciascuno in considerazione in modo ugualitario. In questo algoritmo non capita infatti che un nodo $b_i \in A(ID)$ abbia un'importanza diversa da qualcunque altro.

La funzione che implementa questa metrica inizia la computazione prendendo la lista "del vicinato" c_i in ingresso e, per ognuno dei nodi candidati, ottiene il numero delle sue connessioni appartenenti alla rete G1. Si può anche considerare come la dimensione di $A(c_i)$. Il valore ottenuto è il numero di amicizie in comune. Se si va a guardare la situazione dalla parte opposta, ovvero dal punto di vista degli m nodi candidati, viene così generata una nuova lista di occorrenze per ognuno di essi.

$$\forall c_i \in S^{ID} : AC(c_i) = |A(c_i) \cap A(ID)|$$

La metodologia degli amici in comune è quella più immediata e anche la più semplice nel caso si debbano implementare dei suggeritori di amicizia. Questo perché, per ogni possibile nodo candidato, maggiore è il numero di amici in comune e più alta è la probabilità di conoscere o aver sentito parlare di tale persona e che comunque non la si abbia ancora aggiunta per un

qualsiasi motivo. È possibile ad esempio che la si sia incontrata in qualche conferenza o attività, oppure che sia un persona con cui si è seguito un corso universitario o ancora un ex compagno delle Scuole Elementari che condivide altre amicizie, e così via.

Amici numerosi e legami deboli Si capisce presto come la tipologia degli amici in comune sia di impatto e possa incidere anche sulla relativa qualità dei suggerimenti. Chiarendo meglio questo concetto, si può prendere in considerazione un classico esempio di un nodo che ha un numero alto di connessioni nella sua lista amici: l'analisi empirica di persone con un numero elevato di amicizie porta nella maggior parte delle volte alla conclusione che molte non siano reali, ma solamente create "online" e frutto di un aggiunta spesso casuale di persone che proprio non si conoscono, magari con il mero scopo solamente di aumentare il numero di connessioni nella propria lista. Una lista numerosa spesso inidca che ci sono legami più deboli rispetto a una con numero inferiore, ma che presenta sicuramente dei legami reali. Per questo motivo si è pensato di dare un peso adeguato a questo legame con la prossima metrica, concedendo credito al numero di amicizie che ogni nodo $b_m \in P^{ID}$ possiede.

3.2 Amici in Comune Normalizzato

Questa seconda metrica è una variante della prima. Essa è stata pensata con lo scopo di dare vantaggio nella classifica di suggerimento ai nodi candidati che possiedono un vicinato con caratteristiche differenti rispetto ad altri. Con questo si intende che, se un nodo candidato è collegato a un amico in comune con ID che possiede meno amici rispetto un altro, allora quello con meno amicizie darà un contributo maggiore.

L'algoritmo parte prendendo in input la lista del vicinato per ogni nodo candidato e va a considerare questa volta non solo il numero effettivo degli amici in comune per ognuno ma, al presentarsi di ogni occorrenza, ne calcola un valore *normalizzato* rispetto al numero di amici totali del nodo in comune. Eccone una formalizzazione:

$$\forall c_i \in S^{ID} : ACN\,(c_i) = \sum_{b \in A(c_i)} \frac{1}{\log(A(b)+1)}$$

In questa formalizzazione, come si sarà notato, compare l'inversa della funzione logaritmica $f(x) = \frac{1}{log(x+1)}$. È chiaro che $x + 1 = 2$ non potrà mai verificarsi, perché un nodo $i \in A(ID)$ sicuramente possiede più di due amici (ID e c_i).

Per spiegare e dare una possibile lettura dell'utilità dei due algoritmi e per essere più comprensibili, si può introdurre in modo pratico l'esempio seguente.
Si considerino due nodi candidati c_i, con 2 e 3 amici in comune con ID.

Le liste di amici dei due nodi sono formate rispettivamente dal nodo “3” (che ha 37 amici) e da “ 7” (53 amici), e da “6” (con 215 amici), “8” (416 amicizie) e “9” (con 389 connessioni). Formalizzando:

- $AC\left(c_{25}\right) = \left|A\left(c_{25}\right) \cap A\left(ID\right)\right| = 2$
- $AC\left(c_{34}\right) = \left|A\left(c_{34}\right) \cap A\left(ID\right)\right| = 3$
- $c_{25} = \{3, 7\}$
- $c_{34} = \{6, 8, 9\}$
- $\left|A\left(3\right)\right| = 37$
  $\left|A\left(7\right)\right| = 53$
- $\left|A\left(6\right)\right| = 215$
  $\left|A\left(8\right)\right| = 416$
  $\left|A\left(9\right)\right| = 389$
- $ACN\left(c_{25}\right) = 1.21$
- $ACN\left(c_{34}\right) = 1.19$

Si può immediatamente notare che con il primo algoritmo il nodo candidato c_{34} ha ottenuto un ranking più alto di c_{25}, mentre utilizzando l’algoritmo normalizzato ha ottenuto un valore più basso. Questo è un caso tipico in cui normalizzando si ottiene un risultato che è l’opposto di quello ottenuto senza normalizzazione.

Il normalizziare lo si può anche estendere per le amicizie dei nodi suggeriti: in altre parole, oltre al numero di amicizie del nodo appartenente al grado G1 si può coinvolgere anche quelle del nodo candidato al suggerimento di amicizia, in modo che possa contribuire, per esempio, da moltiplicatore,

sempre favorendo i nodi con meno amicizie totali.

Eccone una formulazione che corregge la precedente:

$$\forall c_i \in S^{ID} : ACN\left(c_i\right) = m \sum_{b \in A(c_i)} \frac{1}{\sqrt{\log(A(b)+1)}}$$

con:

- $m = \frac{1}{\sqrt{\log(A(c_i)+1)}}$ elemento ancora logaritmico per garantire una successiva normalizzazione

Tale estensione non è ancora stata implementata perché prevede che si sia a conoscenza del numero di amici del nodo c_i, che è esterno alla rete G1. Tale dato non è ancora disponibile perché si dovrebbe andare a reperire le amicizie di ogni c_i. Questo dilaterebbe a dismisura il tempo di attesa che ogni volta si presenterebbe per poter generare la rete G2.

3.3 Centralità degli Amici in Comune

In questa metrica ci si è soffermati nell'analizzare la *Vertex Betweenness*, che è una caratteristica strutturale calcolata per ciascun nodo della rete G1 (v. paragrafo 2.1.1). Con questa metodologia si mettono in primo piano tutti quei nodi candidati che possiedono amicizie condivise molto centrali con il nodo analizzato. Con l'aggettivo *"centrali"* si vuole intendere quella caratteristica di un nodo nell'essere un punto di passaggio verso il resto della rete.

L'algoritmo parte prendendo in input la lista del vicinato di ogni nodo candidato e, per ogununo, dà una valutazione quantitativa che somma la *betweenness* di ogni nodo del vicinato. Una volta ottenuti tutti i valori, essi verranno poi ordinati in una classifica decrescente. Per il calcolo di questa metrica è stata utilizzata la seguente formalizzazione:

$$\forall c_i \in S^{ID} : CENTR(c_i) = \sum_{b \in A(c_i)} BET_b$$

con:

- BET_b valore di betweenness calcolato del nodo b

Anche in questa metrica è possibile pensare a alcune modifiche teoriche aggiuntive, per dare un peso anche al nodo candidato e non solo soffermandosi sulla valutazione all'interno della rete G1.
Un esempio potrebbe essere quello di andare a calcolare il valore della vertex betweenness $BETc_i$ anche del nodo candidato c_i, prendendo in considerazione il suo calcolo all'interno di una nuova rete *G2 estesa*. Essa è basata

su G2, che viene però completata con le amicizie fra i nodi che compongono G2. $BETc_i$ la si può sfruttare come ulteriore addendo alla legge iniziale:

$$BET_{c_i} + CENTR\,(c_i)$$

Una seconda idea potrebbe essere quella di calcolare la betweenness BET_{c_i} del nodo candidato c_i immaginando che si trovi all'interno della Personal Network. Si può confrontare così quanto ogni singolo nodo c_i possa essere più "incisivo" rispetto agli altri.

3.4 Cammini Disgiunti

L'idea di fondo di questa metrica è quella che, se un nodo candidato c_i viene raggiunto da più cluster della Personal Network, allora esso dovrà risultare sicuramente più interessante e a sua volta gradito. Questo perché, se se collegato a più gruppi, il candidato, probabilmente conoscendo persone differenti già raggruppate tra di loro, può essere un utente che ha diversi aspetti comuni con il nodo centrale stesso. Facendo un esempio pratico, se una rete è suddivisa in tre grossi cluster, come per esempio quello che comprende i compagni della squadra di calcio, quello dei compagni universitari e quello dei giovani del proprio quartiere, se c'è un potenziale nodo che viene a collegarsi ai tre cluster, sarà molto più facile che il nodo centrale possa ritenere molto interessante aggiungerlo alla lista dei propri amici, piuttosto che magari andare a incrementarla con un utente differente, magari che abita vicino nel nostro quartiere, ma non condivide altri interessi. Nella classifica di suggerimento quindi sarà più svantaggiato, rispetto quello con più cluster condivisi, perché ha pochi aspetti comuni con la persona analizzata.

Ora si possono introdurre delle notazioni per poter comprendere meglio i concetti:

$$CL = \{N_1,\ N_2, \ldots,\ N_k\}, \text{ suddivisione in cluster della rete G1}$$

dove:

$$N_1 \cup N_2 \cup \ldots \cup N_k = N(G1), \text{ tutti i nodi della rete G1}$$

Le metrica prevede che si cominci la computazione prendendo in input la lista del vicinato di ognuno dei nodi candidati c_i e la suddivisione in cluster (CL) della rete G1. Si andrà così ad analizzare ogni nodo in comune b_n, che sta tra il nodo centrale e quello candidato, e che sarà componente effettiva di un singolo cluster. Analizzando tale lista dei nodi in comune, l'algoritmo prevede di contare il numero di cluster differenti che portano dal nodo centrale al nodo candidato passando per i b_n. Per rendere quanto più rigorose le parole, eccone una formulazione che regola la metrica:

$$\forall c_i \in S^{ID} : CD\,(c_i) = |Z|$$

dove:

$$Z = \{N_i \in CL \ t.c. \ b \in N_i, \ b \in A(c_i)\}$$

In questo algoritmo la classifica viene generata in modo decrescente: i primi nodi candidati che la compongono vedono passare i cammini contemporaneamente tra più cluster, mentre quelli posizionati più indietro hanno meno occorrenze.

3.5 Cammini Raggruppati

Anche questa metrica si basa si concentra sulla suddivisione in comunità CL della rete G1, che viene passata in input all'algoritmo. Oltre a ciò un secondo parametro è la lista del vicinato di ogni nodo candidato c_i. A differenza della precedente, la metrica andrà a fare la somma normalizzata del numero di cammini, tra il nodo centrale e il candidato, che transitano per ogni cluster. In questo modo si dà un grosso peso a quanti più "passaggi" ci sono per ogni cluster riconosciuto. Formalizzando:

$$\forall c_i \in S^{ID} : CR\,(c_i) = \textstyle\sum_{i \in CL} log\,(v_i + 1)$$

dove:

$$v = \{b \;\; t.c.\; b \in N_i,\; b \in A(c_i)\}$$

Sempre tenendo conto che:

$$CL = \{N_1,\; N_2, \ldots,\; N_k\}, \text{ suddivisione in cluster della rete G1}$$

dove:

$$N_1 \cup N_2 \cup \ldots \cup N_k = N(G1), \text{ tutti i nodi della rete G1}$$

Questa seconda procedura potrebbe risultare in qualche aspetto simile a quelle che considerano gli amici in comune, e che non utilizzano il clustering. Potrebbe così capitare che alcuni suggerimenti possano venire sovrapposti, soprattutto se si considerano reti non troppo estese.

La classifica viene generata ancora una volta con carattere decrescente, lasciando nelle posizioni più alte i nodi candidati che hanno più "linee di collegamento" verso i differenti cluster.

Capitolo 4

Implementazione

In questo capitolo si analizzeranno tutte le parti implementate del progetto, approfondendo le varie librerie utilizzate per la *raccolta dei candidati*, per la *costruzione della Personal Network* e dei *Ranking* che generano infine i *Suggerimenti*.

API e loro limiti Le API, acronimo che sta per *Application Programming Interfaces*, sono delle interfacce che Facebook ha messo a disposizione degli sviluppatori per poter accedere ai propri dati. Per usufruirne è necessario creare un'applicazione Facebook dall'interno della quale si possono utilizzare tutte le funzionalità offerte. Ci sono due tipi di applicazioni: quelle *Web*, utilizzabili con linguaggi di programmazione "Client-Server"[1], e quelle *Desktop*, utilizzabili con linguaggi di programmazione classici come Java e

[1]Sono un'evoluzione dei sistemi basati sulla condivisione semplice delle risorse. La presenza di un server permette ad un certo numero di client di condividerne le risorse, lasciando che sia il server a gestire gli accessi alle risorse per evitare conflitti tipici dei primi sistemi informatici.

PYTHON[2]. Per poter usufruire dei *Tools* forniti dalle API all'esterno della piattaforma *Developers*[3], bisogna far riconoscere l'applicazione al proprio codice instanziando una variabile *apiKey*, che è la chiave univoca generata da Facebook, e disponendo di una *sessionKey*, che ha il compito di riconoscere chi è autorizzato o meno a utilizzare l'applicazione stessa. Nel caso di versioni Desktop, l'autorizzazione viene effettuata attraverso l'apertura di un *Browser*[4], che dà il via libera solo ad una lista di utenti. Una volta avvenuta questa procedura, è possibile utilizzare tutte le API.

In questo lavoro di ricerca è stato utilizzato un unico metodo, tra i tanti delle API, che permette di controllare se due Facebook ID, identificatore univoco che differenzia ogni persona che ha un account su Facebook, hanno stretto una relazione simmetrica di amicizia:

Metodo "areFriends"

```
friends.areFriends(uids1, uids2)
```

dove *uid1* e *uid2* sono le liste contenenti da 1 a n Facebook ID

Questo metodo richede in input le liste di Facebook ID, mentre i valori ritornati sono a scelta dell'utente in formato *XML*[5] o *JSON*[6] e contengono le informazioni di amicizia sulle coppie di Facebook ID. Ecco un esempio per i due tipi di formati:

[2]È un linguaggio multipiattaforma ad alto livello interpretato, orientato agli oggetti, adatto, tra gli altri usi, per sviluppare applicazioni distribuite, scripting, computazione numerica e system testing.

[3]http://developers.facebook.com/tools.php

[4]Programma per navigare il WWW.

[5]È un metalinguaggio utilizzato per creare nuovi linguaggi, atti a descrivere documenti strutturati.

[6]È un formato adatto per lo scambio dei dati in applicazioni client-server che ritorna elementi di tipo "chiave: valore".

Esempio di XML

```
<friends_areFriends_response>
        <friend_info>
            <uid1>560804804</uid1>
            <uid2>746017555</uid2>
        <are_friends>0</are_friends>
</friend_info>
</friends_areFriends_response>
```

Esempio di JSON

```
{"uid1":560804804,"uid2":746017555,"are_friends":false}
```

Le API sono molto veloci e utili ma diventano instabili nei casi in cui le liste di Facebook ID richiedono tempi di elaborazione della risposta troppo lunghi. Questo è dovuto al numero di controlli da eseguire che alla lunga portano a un *HTTPError*[7] che non permette di arrivare al completamento della risposta della API.

È proprio per questo motivo che dove non riescono ad arrivare le API, abbiamo introdotto l'utilizzo di uno scraper, che sarà approfondito nel prossimo paragrafo.

Scraper e suoi limiti Lo *scraping* è una tecnica di programmazione creata per riuscire a reperire informazioni direttamente dalle pagine WEB, facendo sembrare che la navigazione sia effettuata da un agente umano, con la possibilità perfino di simulare il Browser utilizzato.
Per implementare lo *scraper* si è scelto di utilizzare il linguaggio di programmazione *PYTHON*, nella versione 2.6 (`http://www.python.org`), che

[7]Errore di risposta del protocollo HTTP.

tra le sue peculiarità permette un rapido interfacciamento con le API di Facebook. Questo è permesso dalla libreria *pyfacebook*, nella sua unica versione 0.1 (`http://github.com/sciyoshi/pyfacebook`), che permette di effettuare le richieste e ricevere risposte con i dati provenienti dalle API.
Lo scraper, creato in PYTHON per poter interfacciarsi con le pagine WEB e interpretarne il codice HTML di ciascuna, ha usato la libreria *Mechanize*, nella versione 0.1 (`http://wwwsearch.sourceforge.net/mechanize`), che permette di trasformare la pagina passata tramite un URL[8] all'oggetto *Browser* che la legge e la rende disponibile per l'uso.

Tra i possibili limiti che si incontrano nell'utilizzo di uno scraping è il dover utilizzare dei *timeout* per evitare che le richieste siano troppo veloci rispetto a quanto un agente umano possa esserlo. Nel muoversi quindi tra le pagine di Facebook, si è inserito l'accorgimento di mettere delle piccole pause (dell'ordine del secondo) per rendersi somigliante a una persona normale che naviga nel sito stesso.

[8] È una sequenza di caratteri che identifica univocamente l'indirizzo di una risorsa sul WWW.

4.1 Raccolta dei Candidati

In questa sezione sarà possibile approfondire dettagliatamente come è avvenuta l'implementazione del modulo per la raccolta dei candidati da suggerire.

Come prima cosa si vuole mettere in risalto quali input necessitano e quali saranno i risultati che genererà il modulo. Per quanto riguarda i dati indispensabili perché la computazione possa avere inizio, è necessario che sia conosciuto l'identificatore con cui *Facebook* riesce a differenziare univocamente una persona dall'altra. In output ci si aspetterà due file separati di testo.

Come si deduce dal *Diagramma di flusso* in Figura 4.1, a partire dal valore in input (d'ora in poi *Facebook ID*) viene lanciato lo Scraper, descritto nel paragrafo 4.1, capace di ottenere la lista amici A e di salvarla su disco. Essa contiene tutti i Facebook ID e i relativi nomi degli amici che il nodo in esame possiede. Dopo aver ottenuto A, si passa a esaminare ogni elemento i della sua lista amici B_i: l'implementazione riconosce se tale lista sia presente su disco e, se recente, la carica ottimizzando i tempi; se non presente o obsoleta, viene lanciato nuovamente lo scraping che determina la lista stessa e la salva. Si ottiene così una nuova lista B con tutti gli "Amici di Amici". Il passo successivo è quello di salvare il numero di amicizie NA dei nodi presenti all'interno della lista A, che a sua volta si accorpa a quella B. Ora l'implementazione prevede che vengano eliminate tutte le ripetizioni, per far sì che ogni nodo sia presente una sola volta nella lista definitiva. Da qui si prepara la lista $L2$ per la scrittura su file, sfruttando il formato "*chiave: listaID*". Si abbina ad ogni chiave, contenente un qualsiasi nodo, alla listaID,

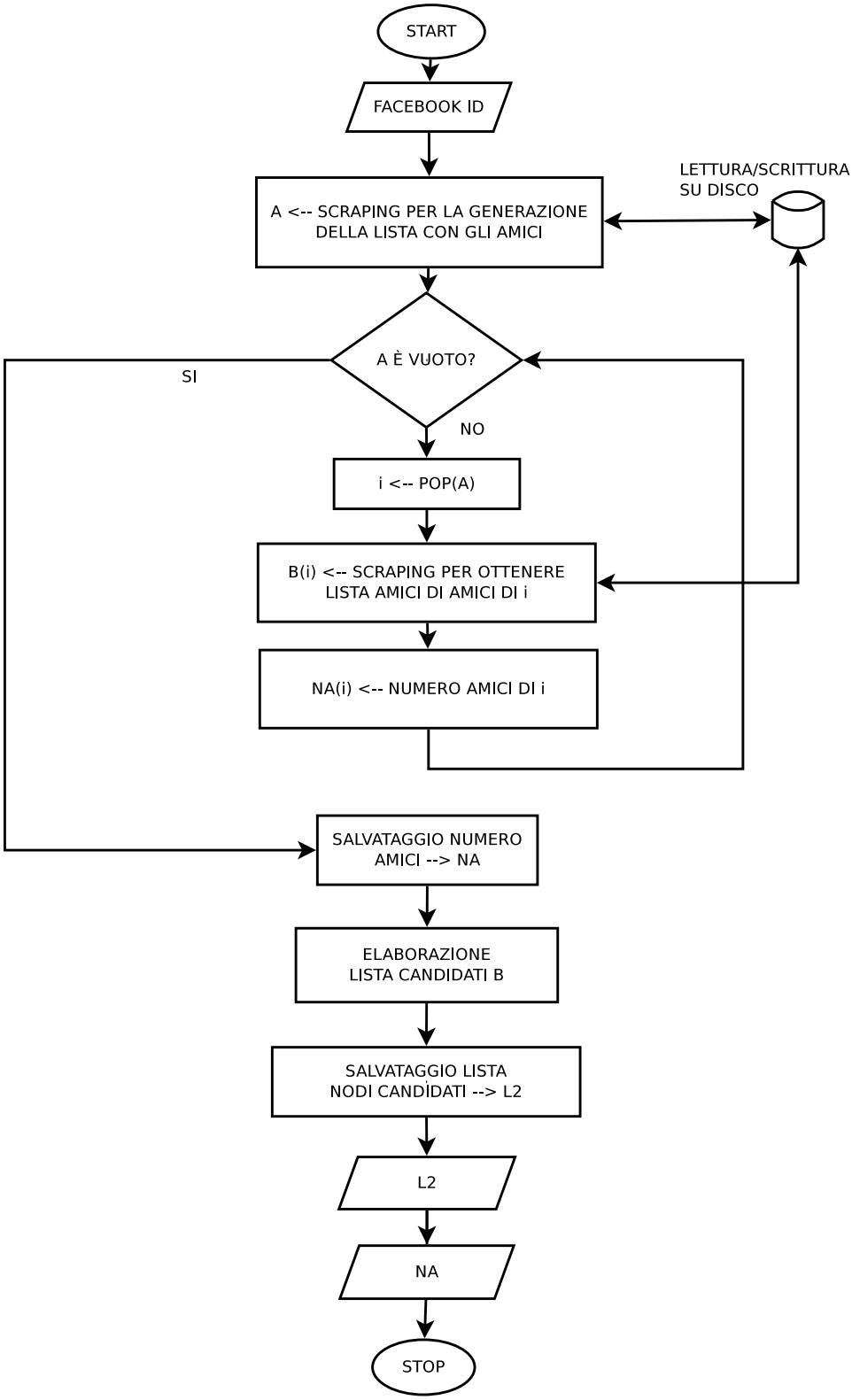

Figura 4.1: Diagramma di flusso che descrive l'implementazione del modulo

contenente i nodi collegati ad esso, formando così un riferimento tra la chiave e gli amici che Facebook ID ha in comune ogni candidato. Ora sia NAche $L2$ sono pronte per essere scritte su disco come output del modulo.

Scraping Pagina Amici La pagina su cui Facebook elenca tutte le connessioni di amicizia di un singolo utente viene caricata a questo URL:

`www.facebook.com/friends/?id=12345678`

dove il numero finale dopo il carattere "=" corrisponde al Facebook ID della persona.

Sfruttando questa pagina, si può risalire a tutte le informazioni necessarie al modulo. C'è però un limite, che sta nel visualizzare solamente 400 amici in ogni caricamento: questo inconveniente è stato aggirato utilizzando un ciclo che permette di parametrizzare l'ultima parte di URL e di accedere ai gruppi di 400 amici successivi, sempre che siano presenti.

La seguente parte di pseudocodice permette di salvare in memoria i dati necessari, partendo proprio dall'URL parametrizzato da ID:

Pseudocodice dello scraper

```
i = 0


while(trovato)
begin
  page = [...]?id=ID&s=(i*400)"
  i = i+1
  temp = esprRegolareID
```

```
  nome = esprRegolareNome
  if (len(temp) == 0) then
        trovato = False
  else
        amici = amici + temp
  end if
end while

amici = nome + amici
```

Un'altra peculiarità della procedura è che i Facebook ID degli amici più i loro nomi vengono riconosciuti all'interno del codice della pagina attraverso l'utilizzo di una *espressione regolare*. Essa permette di essere certi che i dati cercati siano univoci e non ripetuti:

Espressione Regolare 1

```
'friendClick(this, event, (\d+).*?fname..>(.*?)<'
```

dove il contenuto tra tonde conterrà il valore sulla pagina:

1. (\d+)

   determina il Facebook ID

2. (.*?)

   determina il nome associato all'account

I valori ritornati vengono così salvati nella lista amici e, nel caso, scritti su disco.
Attraverso un'altra *espressione regolare* viene letto il numero di amici:

Espressione Regolare 2

```
'ha (\d+) amici'
```

e viene filtrato nel momento in cui la lista va oltre le 1200 occorrenze. Questo per evitare, come descritto nel paragrafo 3.1, la situazione che porterebbe a prendere in considerazione nodi con legami di amicizia troppo deboli.

4.2 Costruzione della Personal Network

In questa sezione sarà possibile capire come è stato possibile ideare il modulo che si occupa della generazione della Personal Network corrispondente al Facebook ID.

Il modulo prende in input il Facebook ID e attraverso la procedura di scraping, descritta sempre nel paragrafo 4.1, si ottiene la lista A degli amici di ID.

Prendendo ogni nodo dalla lista A si procede così all'elaborazione della Personal Network. Attraverso l'uso della libreria che si interfaccia con le API, si interroga Facebook in modo da ottenere "chi è amico di chi" grazie alla funzione *areFriends*, già descritta prima. Vengono così costruite le due liste dei Facebook ID, di cui questa funzione ha bisogno come parametri, e si ottiene come output quali coppie di nodi sono amici tra loro e quali non lo sono. Filtrando chiaramente soltanto i nodi che hanno un legame di amicizia tra loro, si costruisce la struttura dati che contiene n righe quanti sono i nodi b_n di A. In ogni riga i è presente una stringa che contiene solo i nodi che hanno un legame con b_i.
In questo momento della computazione, la matrice è utile per generare la rete in formato Pajek (`http://vlado.fmf.uni-lj.si/pub/networks/pajek`), che è un tipo di file con estensione *.net* che permette di rappresentare, mediante un file di testo, la Personal Network (si veda la Figura 2.1).

Per chiarire l'output che viene generato dal modulo, si scrive qui di seguito un esempio esplicativo di file in formato Pajek:

*Vertices 34472

1 1245742060

2 1474010195

3 559269642

...

*Edgeslist

1 2 6 18 21 30 136 159 160

2 1 6 22 30 38 49 78 82 92 128 136 157 167

3 7 72 107 137 158

...

dove:

- *Vertices 34472 descrive che la rete è formata da 34472 nodi
- 1 1245742060, il nodo 1 è identificato dall'ID 1245742060
- ...
- *Edgeslist introduce la lista delle connessioni
- 1 2 6 18 21 30 136 159 160, il nodo 1 è collegato con i nodi 2, 6, 18, 21, 30, 136, 159, 160
- ...

L'output sarà quindi la Personal Network in formato Pajek, che sarà pronta per essere analizzata dai singoli moduli e determinarne i differenti Ranking, descritti nella prossima sezione.

4.3 Costruzione dei Ranking

In questa sezione approfondiremo la descrizione dell'implementazione di ciascuno degli algoritmi utilizzati per generare le classifiche di Suggerimento.

Per eseguire l'analisi del calcolo della *Betweenness* e della suddivisione in *Cluster* sulla rete G1 è stata utilizzata la liberia *igraph* per PYTHON, nella versione 0.5.3-6 (`http://igraph.sourceforge.net`). Igraph è un pacchetto creato per riuscire a manipolare grafi diretti e indiretti e possiede implementazioni classiche di teoria dei grafi e anche algoritmi per la ricerca delle comunità.

4.3.1 Amici in Comune

Il *Diagramma di Flusso* in Figura 4.2 mette in evidenza, come già ampiamente discusso nella sezione 3.1, la semplicità di implementazione di questo modulo. Esso prende in input la lista $L2$ e per ogni sua riga, che rappresenta ogni nodo candidato c_i va a conteggiare il numero di occorrenze trovate, che sono le amicizie in comune. Da qui inizia il procedimento, identico per tutte le metriche, che che genera una classifica decrescente e ne rappresenta i ranking: si controlla se ci sono ripetizioni e, dopo aver creato tale set di s valori decrescenti, si dimensiona la lista *classifica* che sarà quella in output della procedura. Per riempire classifica si utilizza un ciclo annidato di ricerca che, per ogni elemento di contributi, posiziona quest'ultimo nella corretta posizione di classifica.

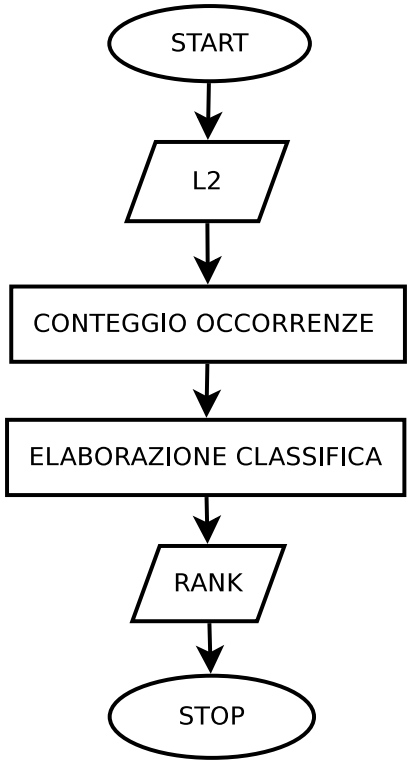

Figura 4.2: Diagramma di flusso che descrive l'implementazione del modulo

4.3.2 Amici in Comune Normalizzato

Il *Diagramma di Flusso* relativo a questo modulo è in Figura 4.3 e, a differenza del precedente, prende in output oltre alla lista $L2$ dei nodi candidati (*neighbors* nello pseudocodice) anche quella NA (*numeroAmici* nello pseudocodice), che contiene il numero di amicizie di ognuno dei nodi appartenenti alla rete G1. L'elaborazione su $L2$ prevede di eseguirne la scansione di ogni nodo candidato c_i e di ottenerne tutti i nodi b_n. Per ogni b_n si va a prelevare il numero di amicizie relativo dalla lista e si esegue la somma totale normalizzata, come descritto dalla seguente procedura:

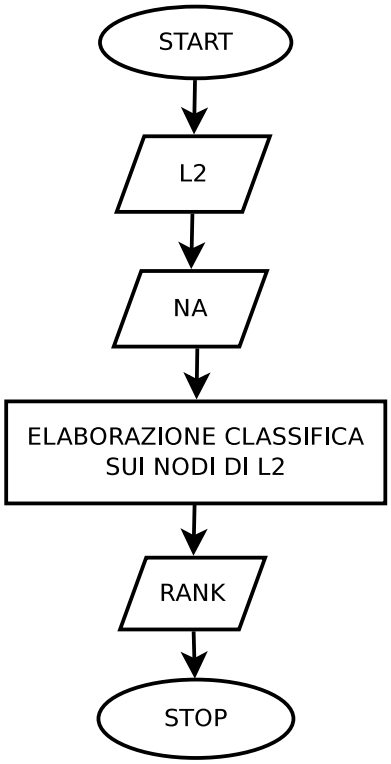

Figura 4.3: Diagramma di flusso che descrive l'implementazione del modulo

Pseudocodice Analisi Somma Normalizzata

```
for i:=0 to len(neighbors) do
begin
  val=0
        for j:=0 to len(neighbors[i])
        begin
      if (neighbors[i][j] in (0,v1)) then
        valore = numeroAmici[neighbors[o][p]]
        if valore != 0 then
          val = val + (1/log(valore+1))
        endif
      endif
```

```
        end
  salva val
end
```

Per ogni candidato si ottiene così un valore val_i che verrà salvato insieme a tutti quelli dei candidati in una lista che sarà poi passata poi alla procedura che genera la classifica decrescente.

Un possibile limite Si propone ora la possibilità futura di suggerire i nodi candidati se e solo se possiedono un numero di amici inferiore ad una soglia prefissata: tale implementazione non è ancora stata effettuata perché prevede che si sia a conoscenza del numero di amici di un nodo esterno alla rete G1. È un dato però non ancora disponibile perché è necessario un tempo elevato per reperire tale informazione utilizzando uno scraper, quindi diventa un limite da non sottovalutare. Tale limite si potrebbe superare se si andasse a determinare il numero di amici solamente dei primi "potenziali" candidati da suggerire: questo fermerebbe la ricerca entro al massimo le 50 unità, contro le possibili decine di migliaia se si dovessero cercare su tutti i possibili nodi candidati appartenenti alla rete G2. È chiaro però che in questo modo si andrebbe a unire scraping e analisi, che attualmente sono in due passi sequenziali.

4.3.3 Centralità degli Amici in Comune

In questa metrica, schematizzata nel *Diagramma di Flusso* in Figura 4.4, si è previsto che il modulo abbia un doppio input: la lista $L2$ dei nodi candidati (*neighbors* nello pseudocodice) e la rappresentazione del grafo della rete $G1$. Proprio su quest'ultima viene calcolata la *Vertex Betweenness* per tutti i b_n

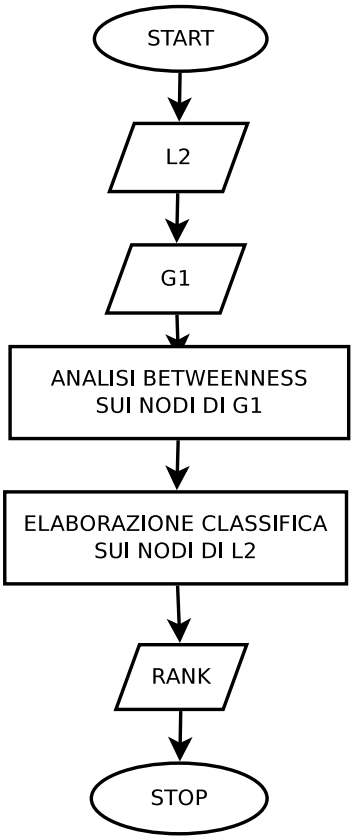

Figura 4.4: Diagramma di flusso che descrive l'implementazione del modulo

nodi. Ogni valore verrà poi utilizzato per il calcolo della punteggio totale di ogni nodo candidato c_n, prendendo in esame tutti i nodi del vicinato b_i. Una volta creata la lista con i valori di ogni candidato, essa verrà passata alla procedura che potrà stilare la Classifica decrescente.

Ecco la procedura per la computazione dei valori:

Pseudocodice Analisi Betweenness

```
for i:=0 to len(neighbors) do
begin
  val=0
  for j:=0 to len(neighbors[i])
```

```
  begin
    if (neighbors[i][j] in (0,v1)) then
          val = val+betweenness[neighbors[o][p]]
        endif
  end
  salva val
end
```

dove:

- la lista *betweenness* contiene i valori calcolati dal metodo *betweenness()* sulla rete G1
- *val* è la somma delle betweenness per ogni nodo candidato c_n

La classifica decrescente costituisce così l'output del modulo.

4.3.4 Cammini

I due moduli che generano il ranking per *Cammini Disgiunti* e *Cammini Raggruppati* funzionano identicamente per i primi passi, salvo differenziarsi solamente nella generazione delle due liste di valori da passare al Generatore per la classifica. Come si nota dal *Diagramma di Flusso* in Figura 4.5, tali moduli prendono in input la lista $L2$ (*neighbors* nello pseudocodice) e la rappresentazione della rete $G1$. Da qui si analizza G1 effettuando il clustering con i metodi disponibili grazie a *igraph*. Gli algoritmi di ricerca di clustering funzionano su reti che presentano solo ed esclusivamente delle *Componenti Connesse*: in altre parole le reti devono avere delle "sotto-parti" che presentano tutti i nodi collegati tra loro e che non hanno nodi separati da tutto

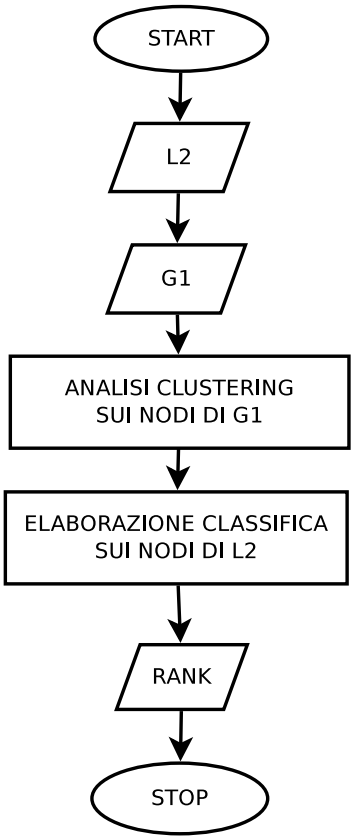

Figura 4.5: Diagramma di flusso che descrive l'implementazione del modulo

il resto. Per scelta, non si è andati ad applicare gli algoritmi nelle sotto-reti con meno di sei nodi, perché già chiaramente formate in un cluster di dimensione sufficiente, ma anche inutile da suddividere. La scelta su quale algoritmo utilizzare tra i quattro incontrati è caduta su quello di *Spin Glass*, perché, durante le prove su varie dimensioni di reti G1, aveva molti aspetti positivi rispetto ai rimanenti algoritmi: questi aspetti sono senz'altro il tempo di computazione che, a parità di rete, era meno elevato per trovare la soluzione e, in secondo luogo, si è sempre riuscito a ottenere un valore di *modularità* più alto rispetto ai concorrenti: come spiegato nel capitolo 2, questo porta ad ottenere una suddivisione in cluster strutturalmente di qualità maggiore, che è l'obiettivo sostanziale di queste metriche.

Descriviamo ora la procedura portante del modulo, introducendo lo pseudocodice che ha come parametri le due liste *clusterchecked* e *clustercount*, che segnalano rispettivamente se un cluster è già stato controllato, per "Cammini Disgiunti", e, per "Cammini Raggruppati", l'occorrenza delle linee uscenti da ogni cluster che raggiungono lo stesso nodo candidato:

Pseudocodice Analisi Cammini

```
for i:=0 to len(neighbors[i]) do
begin
  val=0
  valLog = 0
  if (neighbors[o][p]) in range(v1) then
    for z:=0 to len(clusters) do
    begin
      control = False
      for k:=0 to len(clusters[z]) do
      begin
        if (neighbors[o][p]) in clusters[z][k] then
          if pesato == False then
            if clusterchecked[k]==False then
              val=val+1
              clusterchecked [k] = True
              control = True
              break
            endif
          else:
            clustercount[k]= clustercount[k]+1
            control = True
```

```
            break
          endif
        endif
      if(control==True) then
        break
      endif
  endif
  if pesato == False then
    salva val
  else
    for g:=0 to len(clustercount) do
    begin
          valLog=valLog+math.log(clustercount[g]+1)
        end
    salva valLog
  endif
end
```

dove:

- la lista *clusters* contiene la suddivisione in comunità della rete G1
- *clustercheck* è la lista contenente i flag che controllano se un cluster è già stato attraversato
- *clustercount* è la lista contenente i passaggi attraverso ogni cluster
- *val* è il numero di cammini disgiunti attraverso i cluster che raggiungono ogni nodo candidato c_n
- *valLog* è la somma normalizzata del numero di cammini attraverso ogni cluster che raggiungono ogni nodo candidato c_n

Come per gli altri moduli viene analizzata la lista L2 dei nodi candidati attraverso l'utilizzo di due cicli annidati. Ogni nodo appartenente al vicinato appartiene a un solo cluster e quindi per le due versioni viene aggiornato l'elemento clusterchecked (*True* o *False*) e incrementato clustercount relativo esattamente a quel cluster. Nella versione senza pesi, se un cluster viene esaminato, e quindi il relativo flag clusterchecked messo a True, lo si ignorerà fino al momento in cui si cambierà il nodo candidato. È diverso per la versione pesata, in cui i cluster possono essere esaminati più volte.
Le liste con i contributi vengono riempite secondo le formulazioni viste nel Capitolo 3 e passate al Generatore dei ranking.

4.4 Costruzione dei Suggerimenti

Figura 4.6: Pagina HTML di presentazione dei suggerimenti

Non vengono proposti al nodo analizzato tutti i suggerimenti provenienti dall'output di ogni modulo, ma solamente i primi dieci nodi candidati per ognuno di essi che hanno maggiore peso nel ranking. La lista dei suggerimenti finale che viene creata è ovviamente un set, cioè che non presenta ripetizioni.

Per rendere semplice la visualizzazione dei risultati da presentare, si è pensato di creare un file HTML in cui si passassero una serie di link alle

pagine-profilo di Facebook per ogni candidato suggerito, i cui URL sono simili al seguente:

`http://www.facebook.com/profile.php?id=123456`

Utilizzando il seguente

Tag per l'inserimento di Riferimenti

```
<a href="URL">Nome</a>
```

all'interno della codifica HTML, abbiamo creato la pagina come quella in Figura 4.6. In questo modo l'utente riesce a visualizzare in modo rapido ogni persona suggerita navigando sulla sua pagina-profilo di Facebook e nello stesso momento a compilare facilmente il Questionario per la valutazione, la cui analisi la affronteremo nel prossimo capitolo.

Capitolo 5

Valutazione

In questo capitolo si andrà ad analizzare qualitativamente e quantitativamente il lavoro, focalizzando l'osservazione sulle valutazioni emerse dal Questionario presentato ad ogni persona che ha formato il campione di studio preso in oggetto. Nella sezione *Metodo di Valutazione* approfondiremo come è stato pensato e a chi è stato rivolto il test, mentre in quella successiva, *Risultati*, si porrà l'attenzione sui dati ritornati dalle valutazioni. Nell'ultima sezione invece ci si soffermerà sulle *Considerazioni* finali.

5.1 Metodo di Valutazione

5.1.1 Campione di Test

Il *Campione*[1] per la valutazione del *Suggeritore di Amicizia* è stato scelto in modo completamente casuale tra le persone, tutte più o meno conosciute, che, come requisito fondamentale, abbiano registrato un account di tipo "Persona" su Facebook. Questo è considerato il vincolo principale poiché senza un punto di partenza, come l'account e il relativo *Facebook ID*, non è possibile determinare la Personal Network e poter eseguire quindi nessun tipo di Analisi di rete sul soggetto. Non si sono poste altre restrizioni perché il funzionamento del Suggeritore utilizza solamente dati provenienti da uno studio *Topologico* sulla rete stessa e non tiene in considerazione in nessun modo informazioni personali, interessi e contenuti condivisi dal soggetto.
Tutte e ventitre le persone che sono state sottoposte alla valutazione hanno un'età compresa tra i 22 e i 37 anni e presentano una buona varietà per quanto riguarda l'impiego lavorativo: si va dallo studente universitario al ricercatore, passando per il giornalista, l'informatico e il grafico alle dipendenze di un'azienda tessile.

[1] È una rilevazione utile per determinare uno o più parametri di una popolazione, senza però doverne analizzare ogni suo componente. Si compie quindi una stima sul campione, utilizzando un livello di fiducia, come se fosse un valore relativo a tutta la popolazione.

5.1.2 Presentazione e Questionario

Ad ogni componente del campione si sono inviati, attraverso un messaggio sulla casella di Posta Elettronica personale, due documenti che potessero essere il più possibile intuitivi e di semplice utilizzo. Tutto ciò per dare la possibilità di visualizzare l'elenco con tutti i Suggerimenti di Amicizia personali e contemporaneamente per poter avere una valutazione di ognuno.
Il primo documento è un file *HTML statico* visualizzabile attraverso un qualsiasi Browser che, come spiegato nella sezione 4.4, contiene i Suggerimenti di Amicizia presentati attraverso una serie di collegamenti ipertestuali. Essi puntano alla pagina-profilo di Facebook di ogni persona consigliata e permettono a ogni "tester" di visualizzare immediatamente di chi si tratta. Invece il secondo documento è un file in formato *XLS* che dà la possibilità di compilare in forma tabellare la valutazione di ogni suggerimento. Ogni utente ha avuto il compito di rispondere alle domande proposte compilando ogni cella della Tabella e di rimandare il file salvato per poter così effettuare la valutazione. Le domande proposte sono state le seguenti:

1. "CONOSCI QUESTA PERSONA?"

   Permette di specificare il grado di conoscenza dell'utente intervistato verso ogni candidato suggerito. Utilizzando come feedback le seguenti risposte prefissate: *a*) Si; *b*) Di vista; *c*) No; si può interpretare in modo obiettivo se il suggerimento generato ha permesso di determinare una persona più o meno conosciuta.

2. "AGGIUNGERAI QUESTA PERSONA?"

Permette di puntualizzare se il suggerimento generato è di un'importanza tale che porta l'utente intervistato a essere nelle condizioni di inviare immeditamente una richiesta di amicizia. Con le risposte: *a*) Si; *b*) Indifferente; *c*) No; si può capire se effettivamente il suggerimento ha sortito o meno il massimo effetto dell'invio.

3. "IL SUGGERIMENTO È INTERESSANTE?"

Dà la possibilità di illustrare se il candidato suggerito ha destato interesse all'utente intervistato. Attraverso le seguenti risposte: *a*) Si; *b*) Indifferente; *c*) No; l'utente esprime il livello di attrazione derivato del suggerimento.

5.2 Risultati

5.2.1 Performance

Per usufruire del Suggeritore di amicizia è necessario sottolineare la rilevanza che possiede il tempo di esecuzione del software necessario e quindi studiare quanto incidono le *Performance* sul suo utilizzo.

L'esecuzione del modulo per effettuare l'Analisi di rete e la costruzione dei Ranking, descritto nella sezione 4.3, e di quello per la costruzione della Personal Network, illustrato nella Sezione 4.2, sono sempre computazionalmente più lenti rispetto a quello relativo alla raccolta dei Candidati (Sezione 4.1). Quest'ultimo è in generale è il più veloce perché utilizza un tempo di computazione dell'ordine di $O(k)$ per ottenere la lista candidati, a partire dai k nodi della Personal Network. È quindi un elemento lineare che è più rapido se riferito a uno quadratico, come ad esempio $O(k^2)$, che è legato al caso peggiore per ottenere in output la generazione della Personal Network. Quest'ultimo è praticamente equivalente a quello progettato per la generazione dei rank: l'algoritmo deve andare a leggere la lista dei nodi appartenenti al vicinato di ogni candidato e poi assegnare a ognuno un valore euristico. Il tempo di computazione, riferendosi sempre al caso peggiore, sarà ancora quadratico e dell'ordine di $O(n * m)$ (con m numero di candidati e n numero massimo di amici in comune).
Per ottenere la lista dei Suggerimenti di amicizia si dovrà attendere così un tempo massimo di computazione che sarà dell'ordine di:

$$t_{COMP} \begin{cases} O\left(k^2\right) & \text{se k } > \mid \sqrt{n * m} \mid \\ O\left(n * m\right) & \text{altrimenti} \end{cases}$$

5.2.2 Correlazione tra le Metriche

Per dare una reale quantificazione della *Correlazione*[2] tra le metriche considerate, si è andati a effettuarne una comparazione sia sul rank che ciascun candidato occupava nella classifica totale, sia sui valori ottenuti da ognuno.

Il calcolo delle correlazioni si è applicato sugli output ottenuti dagli algoritmi utilizzando il pacchetto *R per Computazioni Statistiche*, nella sua versione 2.9.2 (`http://www.r-project.org`). Sono state effettuate su ogni utente intervistato le correlazioni utilizzando i coefficienti di Pearson[3] e di Spearman[4] partendo dall'input delle liste derivate dalle cinque metriche, arrivando ad ottenere come output un range di valori compresi tra 0 e 1 se ci sono correlazioni dirette, altrimenti tra 0 e -1 se inverse.

Si sono così ottenuti i dati espressi dalle Tabelle 5.1 e 5.2 che esprimono i valori medi delle correlazioni tra le 5 metriche dei 23 utenti, rispettivamente sui rank (Spearman) e sui valori (Pearson).

[2] È una relazione tra due variabili casuali tale che a ciascun valore della prima variabile corrisponda con una certa regolarità un valore della seconda.

[3] Tra due variabili aleatorie esprime la linearità tra la loro covarianza $\sigma_{xy} = \sum_{i=1}^{n}((x_i - x_M)(y_i - y_M))$ e il prodotto delle rispettive deviazioni standard $\sigma_x = \sqrt{\sum_{i=1}^{n}(x_i - x_M)^2}$ e $\sigma_y = \sqrt{\sum_{i=1}^{n}(y_i - y_M)^2}$, con x_i e y_i valori da confrontare e x_M e y_M valori medi.

[4] È un caso particolare del coefficiente di correlazione di Pearson dove i valori vengono convertiti in ranghi prima di calcolare il coefficiente.

Si nota subito, nella prima Tabella, la cella corrispondente alla coppia AC-CR, 0,999, in cui la correlazione è elevata perché le due metriche in questione eseguono una somma, che essa sia normalizzata o meno, sullo stesso gruppo e numero di addendi, quindi il rank di ogni candidato risulta molto simile tra le due metriche in oggetto. Il dato della cella ACN-CENTR, 0,185, al contrario mostra una scarsa correlazione nella disposizione dei ranking perché le due differenti metriche vengono calcolate su valori molto differenti tra loro. Passando in rassegna la Tabella che si riferisce alla correlazione su valori, si nota come "Centralità degli Amici in Comune" e "Cammini Disgiunti" siano poco correlate alle altre quattro metriche, con un contributo medio che si aggira sullo 0,53 e sullo 0,6. Il più basso valore di correlazione si attesta nella cella CENTR-CD, 0,399, ed è dovuto alla grande diversità tra i valori in gioco: per la prima metrica ci sono somme di valori di *Betweenness* che sono diversi dal conteggio di numero di *Clusters* attraversati che Cammini Disgiunti compie nella sua computazione.

	AC	**ACN**	**CENTR**	**CD**	**CR**
AC	1,000	0,631	0,424	0,690	0,999
ACN		1,000	0,185	0,444	0,630
CENTR			1,000	0,320	0,424
CD				1,000	0,718
CR					1,000

Tabella 5.1: Media sulla correlazione di Spearman tra le metriche. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in Comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

Dall'analisi delle Tabelle 5.3 e 5.4 che descrivono l'andamento della *Varianza*[5] emergono ancora dalla 5.3 i valori delle celle ACN-CENTR e AC-

[5]Fornisce una misura di quanto siano vari i valori assunti dalla variabile, ovvero di quanto si discostino dalla media x_M.

	AC	**ACN**	**CENTR**	**CD**	**CR**
AC	1,000	0,992	0,596	0,592	0,907
ACN		1,000	0,583	0,583	0,897
CENTR			1,000	0,399	0,560
CD				1,000	0,818
CR					1,000

Tabella 5.2: Media sulla correlazione di Pearson tra le metriche. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in Comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

CR: essi sono ancora agli estremi opposti, dove il primo 0,036 è il più alto e dà l'idea di quanto ACN e CENTR diano variabilità; mentre per la coppia AC-CR la variabilità è nulla: la somiglianza dei ranking è dovuta al tipo di valori che entrano in gioco nella metrica, che come già detto sono altamente correlati. Se si vanno a valutare i valori della 5.4, si nota immediatamente di come ci sono tre coppie di metriche che hanno varianza tendente praticamente a 0: AC con ACN e CR danno poca variabilità a causa della grossa correlazione sui valori, così come le stesse ACN-CR tra di loro.

	AC	**ACN**	**CENTR**	**CD**	**CR**
AC	1,000	0,013	0,014	0,010	0,000
ACN		1,000	0,036	0,018	0,013
CENTR			1,000	0,017	0,014
CD				1,000	0,011
CR					1,000

Tabella 5.3: Varianza sulla correlazione di Spearman tra le metriche. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in Comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

	AC	ACN	CENTR	CD	CR
AC	1,000	0,000	0,034	0,017	0,001
ACN		1,000	0,037	0,017	0,001
CENTR			1,000	0,029	0,027
CD				1,000	0,008
CR					1,000

Tabella 5.4: Varianza sulla correlazione di Pearson tra le metriche. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in Comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

Sovrapposizione delle Metriche

Si affronterà ora la questione di quanto le metriche producano Suggerimenti ripetuti e quindi in che misura diano output sovrapposti (*Overlapping*). Nell'implementazione si è scelto di considerare solamente i migliori dieci candidati da suggerire per ogni metrica e quindi se ne potranno avere dai 50 del miglior caso fino ad arrivare agli stessi 10 ripetuti da tutte le metriche.

Nella Tabella 5.5 sono stati inseriti i valori medi dell'overlapping scaturiti sulle Analisi dei 23 utenti. Come si può notare le due metriche che sfruttano gli Amici in Comune generano un numero elevato di intersezioni, dato che sorpende relativamente perché ben si sa che lavorano sulla stessa "base". Passando in rassegna i risultati delle restanti metriche, si può notare che il numero totale di sovrapposizioni più basso è relativo a "Cammini Disgiunti" (8,2).
Questi numeri portano già a fare una considerazione di carattere qualitativo sulla funzionalità degli Amici in Comune: il fatto che l'overlapping sia così elevato significa che la metrica in se è buona, perché anche altre arrivano spesso a determinare gli stessi candidati. Le metriche come "Centralità degli Amici in Comune" e "Cammini Raggruppati" danno valori intermedi,

mentre "Cammini Disgiunti" tende sempre a determinare possibili contatti differenti in maniera sistematica con il resto delle metriche, ma soprattutto con quelle basate sugli amici in comune.

	AC	**ACN**	**CENTR**	**CD**	**CR**
AC	10,000	8,565	3,391	1,130	3,739
ACN		10,000	3,826	1,478	3,478
CENTR			10,000	2,913	2,522
CD				10,000	2,652
CR					10,000

Tabella 5.5: Valori medi di Overlapping tra le metriche. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in Comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

Importanza di "Amici in Comune" Per approfondire il discorso dell'overlapping tra le metriche, si è andati a determinare il numero dei Contatti suggeriti non proposti dalla metrica "Amici in Comune". Questo perché essa oltre a essere semplice da calcolare, non ha nemmeno performance peggiori delle altre: la media eseguita sul totale, sempre riferendosi al Campione dei 23 utenti, ha portato a 18,31 nodi suddivisi nelle quattro restanti metriche. Ciò siginifica che è circa il 65% del totale medio, considerato che la prima genera comunque 10 nodi, ed è troppo lontano dall' 80% che si otterrebbe in caso di perfetto equilibrio e senza sovrapposizione tra le differenti metriche. Approfondendo il discorso su ciascuna metrica, l'apporto medio di ciascuna è stato così suddiviso:

- 1,50 per Amici in Comune Normalizzata
- 5,42 per Centralità degli Amici in comune

- 6,17 per Cammini Disgiunti
- 5,25 per Cammini Raggruppati

5.2.3 Valutazione dei Suggerimenti

Se nella sotto-sezione precedente abbiamo dato una valutazione sulla correlazione a coppie delle cinque metriche del progetto, in questa andremo ad approfondire come gli utenti intervistati hanno espresso il loro giudizio sui Suggerimenti di Contatti proposti, valutandone ancora una volta statisticamente le risposte ottenute.

Analisi della Varianza

Le prossime tre Tabelle 5.6, 5.7 e 5.8 illustrano, con valori aggregati medi, il risultato proveniente dalle tre Risposte di Valutazione, suddiviso per singola metrica, alle domande sottoposte al Campione dei 23 utenti.
Quello che maggiormente interessava in questo lavoro di ricerca era lo studio sui *valori affermativi* delle risposte. Prendendo subito in esame le tabelle indicate sopra, si può vedere immediatamente che c'è poca *variabilità* tra le metriche nella Domanda 2 e nella Domanda 3, con uno scostamento rispettivamente di 0.82 e di 0.26, mentre la differenza nella Domanda 1 è molta (2.96).
Proprio a causa di questo valore elevato, si deve introdurre il concetto di *Analisi della Varianza*, o *ANOVA*, che consiste nel verificare la variabilità delle medie tramite test. Si deve provare la seguente ipotesi nulla:

$H0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$

o se non verificata, l'ipotesi alternativa:

$H1$: almeno una media è diversa dalle altre

La verifica di ipotesi si basa sulla determinazione del *p-value*[6]: valori inferiori al livello di significatività prescelto α indicano che sotto $H0$ il risultato osservato è atipico, cosa che porta quindi al rifiuto di $H0$. Questo non significa che tutte le medie siano tutte differenti l'una dall'altra, ma che ci sia almeno una coppia di medie con valori diversi tra loro. È quindi necessario individuare quali siano queste coppie, procedendo nel verificarne l'uguaglianza. La serie di verifiche si ottiene compiendo opportuni test, definiti come "test *Post-Hoc*".

	Si	**Di Vista**	**No**
AC	5,478	2,174	2,348
ACN	5,304	2,087	2,609
CENTR	4,261	2,174	3,565
CD	2,522	2,261	5,261
CR	4,217	2,565	3,261

Tabella 5.6: Media delle Risposte alla Domanda 1. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

[6]Misura la probabilità di estrarre campioni caratterizzati da un valore della statistica F più elevati di quello osservato per il campione in esame.

	Si	**Indifferente**	**No**
AC	2,391	2,478	5,130
ACN	2,391	2,609	5,000
CENTR	2,261	1,609	6,043
CD	1,565	1,870	6,478
CR	2,261	2,261	5,391

Tabella 5.7: Media delle Risposte alla Domanda 2. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

	Si	**Indifferente**	**No**
AC	2,348	2,957	4,696
ACN	2,435	3,217	4,348
CENTR	2,261	3,130	4,609
CD	2,000	2,565	5,522
CR	2,348	3,261	4,304

Tabella 5.8: Media delle Risposte alla Domanda 3. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

Test Post-Hoc

I test *Post-Hoc* permettono di determinare almeno una coppia di medie la cui differenza è considerata significativa. Si procede nel verificare su tutte le coppie possibili di medie μ_s, μ_t l'ipotesi nulla:

$H0$: $\mu_s = \mu_t$

in opposizione a quella alternativa:

$H1$: $\mu_s \neq \mu_t$

a un livello di significatività α fissato.

In questa valutazione abbiamo utilizzato i test LSD (*Least Significance Difference*) di *Fisher* e HSD (*Honestly Significance Difference*) di *Tukey*. La loro differenza fondamentale sta nella statistica utilizzata, che per il primo è la t di Student mentre nel secondo si utilizza la Q studentizzata:

$$LSD = t_{\alpha/2,df}\sqrt{\frac{2S^2}{n}}$$

$$HSD = Q_{\alpha,k,df}\sqrt{\frac{2S^2}{n}}$$

Bisogna quindi provare matematicamente se:

$$\mid \mu_s - \mu_t \mid \begin{cases} > LSD\ (o\ HSD) & \text{se la coppia di medie è significativa} \\ < LSD\ (o\ HSD) & \text{altrimenti} \end{cases}$$

Se si ragiona invece con il p-value:

$$p - value \begin{cases} < \alpha & \text{se la coppia di medie è significativa} \\ > \alpha & \text{altrimenti} \end{cases}$$

Applicando i due tipi di test sui dati di valutazione del nostro campione, abbiamo ottenuto i risultati illustrati nelle Tabelle 5.9 e 5.10. Considerando che i valori calcolati sono $LSD = 1,101$ e $HSD = 1,541$, la colonna "Coppia Significativa" permette di individuare velocemente quale coppia è *Statisticamente Significativa* e quale non la è.

	Diff.	**p-value**	**Coppia Significativa**
AC-ACN	0,174	0,755	No
AC-CENTR	1,217	0,031	Si
AC-CD	2,957	10^{-6}	Molto
AC-CR	1,261	0,025	Si
ACN-CENTR	1,044	0,063	No di poco
ACN-CD	2,783	10^{-6}	Molto
ACN-CR	1,087	0,053	No di poco
CENTR-CD	1,739	0,002	Si
CENTR-CR	0,044	0,938	No
CD-CR	1,696	0,003	Si

Tabella 5.9: Test Post-Hoc LSD. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

Il test di Fisher è meno stringente di quello di Tukey perché LSD è di valore più basso rispetto a HSD. Infatti quest'ultimo determina quattro coppie come significative (tutte le combinazioni delle metriche con quella "Cammini Disgiunti"), mentre Fisher ottiene le stesse coppie più altre due significative ("Amici in Comune" con "Centralità degli amici in comune" e

	Diff.	p-value	Coppia Significativa
AC-ACN	0,174	0,998	No
AC-CENTR	1,217	0,191	No
AC-CD	2,957	10^{-5}	Molto
AC-CR	1,261	0,163	No
ACN-CENTR	1,044	0,336	No
ACN-CD	2,783	10^{-5}	Molto
ACN-CR	1,087	0,295	No
CENTR-CD	1,739	0,019	Si
CENTR-CR	0,044	0,999	No
CD-CR	1,696	0,023	Si

Tabella 5.10: Test Post-Hoc HSD. AC identifica "Amici in Comune", ACN "Amici in Comune Normalizzati", CENTR "Centralità degli amici in comune", CD "Cammini Disgiunti" e CR "Cammini Raggruppati".

con "Cammini Raggruppati"). "Cammini Disgiunti" ha una differenza sulla media molto significativa con quella delle metriche basate sugli Amici in Comune, quindi nonostante abbiano valori medi molto distanti, le metrica "Cammini Disgiunti" è risultata essere significativa e di peso specifico elevato a livello inferenziale. Questo porta ad affermare che questa metrica trova dei Contatti che sono amici nuovi o poco conosciuti.

Per scrupolo si è verificato se anche lo scostamento minimo delle medie per le risposte alle Domande 2 e 3 potesse ottenere un rifiuto dell'ipotesi nulla $H0$ e, quindi, che qualche coppia di medie fosse statisticamente significativa. Effettuato così il test di *Tukey*, esso non ha fatto emergere alcuna coppia significativa per entrambe le risposte. Questo permette di asserire che anche statisticamente tutte le metriche si comportano in modo simile.

5.3 Considerazioni

Rifacendosi all'approfondimento sull'importanza della metrica "Amici in Comune", affrontato nella sotto-sezione 5.2.2, si è deciso di quantificare il numero medio di contatti ritenuti importanti ma non suggeriti da "Amici in Comune", e tra questi andare a scoprirne il livello occupato nella classifica totale di "Amici in Comune" stessa. Come è già stato illustrato, gli amici suggeriti che non fanno parte della lista generata dalla metrica "Amici in Comune" sono complessivamente in media 18,31 ed è stato calcolato l'apporto singolo medio per ogni metrica.

Analizzando le risposte del campione alla Domanda 3 ("IL SUGGERIMENTO È INTERESSANTE?"), si arriva al passo successivo di determinare la parte di questi contatti definiti come interessanti: il valore medio ottenuto è di 2,22 suggerimenti. Si ottiene quindi un 12% che a prima vista darebbe una valutazione di scarsità. Questo in parte è vero ma se si va a riguardare la Tabella 5.8 si può notare che il valore che si assesta attorno alle medie delle Risposte "Si" di ognuna delle cinque metriche. È un confronto che va preso sicuramente con molta attenzione: il calcolo non considera quanto e in che modo i suggerimenti siano sovrapposti, mentre i valori medi delle Risposte per ciascuna metrica tengono invece conto della sovrapposizione. Proprio per questo si farà ora una valutazione di ognuna delle quattro metriche. Nella Tabella 5.12 si riportano in dettaglio le metriche di provenienza dei contatti definiti interessanti dagli utenti intervistati. I contributi medi per metrica sono così distribuiti:

- 0 per Amici in Comune Normalizzati

- 1.26 per Centralità degli Amici in Comune
- 0.91 per Cammini Disgiunti
- 0.61 per Cammini Raggruppati

Si arriva quindi a dimostrare che la reale percentuale da tenere conto non è il 12% uscito dai dati senza distinzione delle metriche, ma invece di considerare come base il rapporto tra i suggerimenti medi interessanti e il valore medio dei suggerimenti non prodotti dalla metrica "Amici in Comune":

- 0% per Amici in Comune Normalizzati
- 23% per Centralità degli Amici in Comune
- 15% per Cammini Disgiunti
- 12% per Cammini Raggruppati

Sul campione intervistato spicca lo 0% della metrica "Amici in Comune Normalizzati" che non passa alcun suggerimento interessante, mentre "Centralità degli Amici in Comune" riesce a trovare sostanzialmente 1 amico su 4 che sia interessante. Un risultato che non è per nulla da sottovalutare.

L'ultimo passo è stato quello di determinare come la parte dei suggerimenti ritenuti interessanti sono stati trattati dalla metrica "Amici in comune". Andando ad analizzare nel dettaglio le graduatorie complete, e quindi oltre la decima posizione, si nota che c'è una grandissima variabilità nei diversi utenti intervistati. Non si può prevedere in alcun modo le posizioni perché sono fin troppo aleatorie: ad esempio l'*Utente5* ha ritenuto interessanti 2 contatti all'esterno dei primi 10 suggerimenti di "Amici in Comune", che sono però tutti nella seconda decina della classifica totale di questa

	Rank AC	Metriche di Provenienza
Utente1	X	Nessuna
Utente2	540	CENTR, CD
Utente3	20	CENTR
Utente4	165	CENTR, CD
Utente5	15	CENTR
	16	CENTR
Utente6	X	Nessuna
Utente7	X	Nessuna
Utente8	13	CR
	34	CENTR,CD,CR
	110	CENTR
	90	CENTR
	401	CENTR
	129	CENTR
	20	CENTR
	62	CR
Utente9	15	CENTR
	210	CENTR
	438	CENTR
	22	CR
Utente10	320	CD
	111	CENTR
	12	CENTR
	96	CD
Utente11	195	CD
	37	CD
	253	CD,CR
Utente12	269	CENTR, CR
	15	CENTR, CD
	499	CENTR, CD, CR

Tabella 5.11: Piazzamenti in AC e metriche di provenienza dei suggerimenti interessanti all'interno della classifica di "Amici in Comune". PARTE1

	Rank AC	Metriche di Provenienza
Utente13	39	CR
Utente14	X	Nessuna
Utente15	X	Nessuna
Utente16	X	Nessuna
Utente17	18	CD
	369	CD,CR
	25	CD
Utente18	58	CENTR
	13	CENTR,CD,CR
	502	CENTR
Utente19	X	Nessuna
Utente20	15	CENTR,CD
	59	CD
	197	CD
	84	CD
	385	CD
	45	CR
	185	CR
	486	CENTR
	560	CENTR
	212	CENTR
	125	CENTR
Utente21	63	CR
	28	CENTR
	233	CENTR
	19	CD
Utente22	89	CD,CR
	435	CD
Utente23	X	Nessuna

Tabella 5.12: Piazzamenti in AC e metriche di provenienza dei suggerimenti interessanti all'interno della classifica di "Amici in Comune". PARTE2

metrica. All'opposto si può notare come l'*Utente22* abbia reputato interessanti contatti che stanno relativamente indietro proprio nella graduatoria di "Amici in Comune".

Capitolo 6

Conclusioni e Sviluppi Futuri

6.1 Conclusioni

Il problema affrontato in questo lavoro di ricerca è stato quello di definire e applicare una metodologia per automatizzare un Sistema di Raccomandazione di Contatti presenti all'interno di una Rete Sociale.
Non trattandosi di un problema così banale, l'unica soluzione è stata quella di suddividerlo in tre sottoproblemi che potessero essere affrontati ognuno in modo differente.
Prima di tutto si è dovuto capire come fossero rappresentati i dati sulla Rete Sociale online e, fattore da non sottovalutare, anche quali di essi potevano essere reperiti o meno. Il primo sottoproblema è stato quello di determinare la lista dei contatti di ogni Utente con un account registrato e attivo sulla Rete Sociale. Si è studiato quindi il metodo migliore per ottenere i dati nel minor tempo di esecuzione possibile. Una volta reperita tale lista, il proble-

ma non era per nulla risolto. Si è dovuto infatti ricercare ogni relazione tra i suoi componenti: questo permette di definire la Personal Network che, oltre a mettere in relazione l'utente analizzato coi suoi amici, determina anche tutte le relazioni che si sono instaurate tra di essi.
Il successivo sottoproblema affrontato è stato quello di andare a recuperare, partendo dai contatti appartenenti alla Personal Network, tutti i possibili candidati ad essere suggeriti. Essi sono stati qualificati come "Amici di Amici" e quindi per ogni componente della Personal Network si è compiuta l'operazione ricorsiva che andasse a cercare la lista dei contatti di ognuno.
In ultimo si è andati ad affrontare il problema centrale di questo lavoro di Tesi, cioè quello che è consistito nel come ottenere una classifica di gradimento una volta ottenuta la lista di tutti i possibili contatti da suggerire. Per questo punto fondamentale sono state scelte cinque differenti metriche che estraessero ognuna un ranking di gradimento.
Le prime due si basano completamente sugli amici in comune, che sono i punti di unione tra l'Utente esaminato e i Contatti potenziali da suggerire. Queste metriche sono semplici da calcolare e hanno buone performance, ma la cosa fondamentale è che spesso, ma non sistematicamente, tendono a rintracciare contatti conosciuti. Il loro svantaggio si è riscontrato nella loro correlazione: si è notata infatti una notevole sovrapposizione, quindi se si dovessero combinare insieme soltanto queste due metriche si otterrebbero risultati molto simili.
La terza metrica, che porta il nome di "Centralità degli Amici in comune", ha messo in luce il vantaggio di dare la priorità a quei Contatti che possiedono amici in comune che più sono un punto di passaggio verso gli altri componenti della Personal Network. Ha però lo svantaggio di penalizzare in modo elevato i nodi che dualmente hanno poca interazione.
La ultime due metriche, chiamate "Cammini Disgiunti" e "Cammini Raggruppati", sfruttano la suddivisione della Personal Network in cluster. La

prima delle due privilegia i potenziali contatti che hanno amicizie appartenenti al maggior numero di cluster, mentre l'altra tende a dare importanza ad ogni cluster proporzionalmente al logaritmo del numero di amici in comune appartenenti ad esso. A differenza delle metriche basate sugli amici in comune, queste spesso tendono a determinare molto frequentemente dei contatti che sono poco conosciuti dall'Utente esaminato. Questo perché la peculiarità alla loro base sta nel suddividere la sua rete in comunità coese: questo porta a raggruppare persone con stessi interessi, ma non necessariamente che debbano essere già conosciute.

Si sono quindi progettati i diversi moduli che hanno affrontato i tre sottoproblemi, per poi crearne un'implementazione dei diversi algoritmi. Si è creata anche un'interfaccia grafica Web che potesse presentare i Contatti ottenuti dalle metriche ai 23 utenti che hanno composto il test di valutazione. I risultati ottenuti hanno portato a ipotizzare ulteriori considerazioni su possibili sviluppi applicabili in un futuro prossimo, come si potrà vedere nella prossima sezione.

6.2 Sviluppi futuri

Il lavoro appena presentato mette in luce alcune possibilità di miglioramento nella scelta di come utilizzare le metriche che analizzano le Reti Sociali.
Siccome non si può parlare di metriche "buone" o "cattive", ma di metriche "diverse" che possono soddisfare esigenze differenti, questo lavoro può essere considerato come un inizio, dove per continuarlo si potrebbe cercare di studiare e caratterizzare diverse classi di utenti introducendo un uso combinato delle metriche stesse. In questo modo si otterrebbe una generazione dei

Suggerimenti che possa essere sotto certi aspetti "orientata all'utente". In altre parole, si potrebbero utilizzare in contemporanea due precise metriche che garantirebbero una maggior soddisfazione nei risultati se utilizzato su un gruppo simile di utenti.

Ad esempio una persona potrebbe gradire solo suggerimento di contatti già conosciuti, quindi una delle metriche che si basa su amici in comune può considerarsi sufficiente. Se però si deve effettuare una raccomandazione a un componente di una rete sociale aziendale, l'orientamento potrebbe essere quello di trovare conoscenze con lo scopo di aumentare il business, grazie a una serie di contatti meno conosciuti. In questo caso le due metriche basate sui cluster oppure combinando "Cammini Raggruppati" e "Centralità degli Amici in Comune" darebbero una resa migliore delle altre.

Un'ulteriore considerazione potrebbe essere quella di chiedere la preferenza direttamente all'utente da esaminare circa un suo interesse maggiore, che può essere il desiderio di scegliere se avere una maggioranza di suggerimenti di contatti conosciuti oppure incogniti. Nel primo caso la metrica preferita sarà ancora una volta "Amici in comune", mentre se la persona preferisse invece vedersi suggerire una lista di Persone sconosciute o quasi, la metrica di cui ci si potrà avvalere sarà sicuramente quella dei "Cammini Disgiunti".

Bibliografia

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[2] Joseph A. Konstan John T. Riedl Badrul M. Sarwar, George Karypis. Application of dimensionality reduction in recommender system – a case study. *WebKDD Workshop*, 1:12, 2000.

[3] Jilin Chen, Werner Geyer, Casey Dugan, Michael Muller, and Ido Guy. Make new friends, but keep the old: recommending people on social networking sites. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 201–210, New York, NY, USA, 2009. ACM.

[4] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70:066111, 2004.

[5] Martin Everett and Stephen P. Borgatti. Ego network betweenness. *Social Networks*, 27(1):31 – 38, 2005.

[6] Santo Fortunato, Vito Latora, and Massimo Marchiori. Method to find community structures based on information centrality. *Phys. Rev. E*, 70(5):056104, Nov 2004.

[7] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215 – 239, 1978-1979.

[8] Linton C. Freeman. Centered graphs and the construction of ego networks. *Mathematical Social Sciences*, 3:291 – 304, 1982.

[9] Herminia Ibarra. Network centrality, power, and innovation involvement: Determinants of technical and administrative roles. *The Academy of Management Journal*, 36(3):479–482, 1993.

[10] D. M. Wilkinson J. R. Tyler and B. A. Huberman. Proceedings of the first international conference on communities and technologies. In E. Wenger M. Huysman and V. Wulf, editors, *FICCT*, 2003.

[11] Peter Killworth. Estimating the size of personal networks. *Social Network*, 12:289–312, 1990.

[12] Yu-En Lu, Sam Roberts, Pietro Liò, Robin Dunbar, and Jon Crowcroft. Size matters: Variation in personal network size, personality and effect on information transmission. In *CSE (4)*, pages 188–193, 2009.

[13] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004.

[14] M.E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39 – 54, 2005.

[15] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Science*, 103:8577–8582, 2006.

[16] M. Sushak P. Bergstrom P. Resnick, N. Iakovou and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Computer Supported Cooperative Work Conf*, 1994.

[17] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks (long version), 2005.

[18] Joerg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74:016110, 2006.

[19] Jörg Reichardt and Stefan Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Phys. Rev. Lett.*, 93(21):218701, Nov 2004.

[20] P.E. Hart R.O. Duda and D.G. Stork. *Pattern Classification*. Wiley Interscience, 2001.

[21] Sam G.B. Roberts, Robin I.M. Dunbar, Thomas V. Pollet, and Toon Kuppens. Exploring variation in active network size: Constraints and ego characteristics. *Social Networks*, 31(2):138 – 146, 2009.

[22] Stanley Wasserman and Katherine Faust. *Social network analysis: methods and applications*. Cambridge, 1994.