Connettività algebrica e ricerca di comunità in grafi di reti sociali

Mele Giampaolo

March 10, 2011

Abstract

Il secondo autovalore della matrice laplaciana di un grafo, detto anche connettività algebrica, è uno degli invarianti più importanti di un grafo, in particolare è possibile fare stime sul diametro e sulla distanza media, decrivere la crescita di un grafo e definire il concetto di comunità. Inoltre l'aspetto più interessante è la rinumerazione dei nodi (automorfismo) che è possibile fare usando il vettore di Fiedler, ovvero l'autovettore della molteplicità algebrica. Verrà infine proposto un algoritmo di tipo divide et impera per la ricerca delle comunità.

Prime definizioni

Un grafo G è definito come due insiemi, un insieme V detto insieme dei vertici e un insieme $E \subseteq V \times V$ detto insieme degli archi, usualmente indicheremo $G = (V, E)$. Un grafo si dice non orientato se $(v, w) \in E$ allora $(w, v) \in E$, ovvero se il fatto che ci sia un arco dal nodo v al nodo w vuol dire che c'è un arco anche da w in v, se questa condizione non vale allora il grafo di dice orientato. D'ora in avanti si considara G un grafo non orientato con V di cardinalità finita n. Definiamo il grado di un nodo v come il numero di nodi che sono collegati a v da un arco, ovvero

$$deg(v) = |\{(v, w) \text{ tale che } w \in V, (v, w) \in E\}|$$

Definiamo cammino tra v e w una successione di elementi di $e_k \in E$ tali che $e_k \cap e_{k+1} \neq \emptyset$, per esteso $(v, v_1), (v_1, v_2), ..., (v_k, v_{k+1}), ..., (v_{n-1}, v_n), (v_n, w)$. Tra tutte le successioni quella con meno elementi sarà detta cammino minimo. Possiamo dunque introdurre la funzione distanza tra due nodi $d(v, w)$ come la lunghezza del cammino minimo. Definiamo dunque il diametro di un grafo come

$$diam(G) = \max_{(v,w) \in V \times V} d(v, w)$$

Esempi

Un primo esempio di grafo può esser quello delle città italiane. Dunque poniamo che V sia l'insieme delle città italiane e c'è un arco tra due città se e c'è una strada diretta che le unisce. Da questo tipo di grafo ci aspettiamo due cose:

che sia sparso, ovvero $|E| << |V|^2$ e che abbia un diametro decisamente grande. Inoltre, come si vedrà dopo, avrà anche una connettività algebrica bassa .

Un altro esempio può esser quello delle reti sociali come facebook. L'insieme dei nodi è costituito dall'insieme degli utenti, c'è un arco tra due utenti se i due utenti sono amici.
Ogni utente su facebook ha mediamente 120 amici, e gli utenti su facebook sono circa 500 milioni. Quindi siamo nelle stesse ipotesi di prima, abbiamo un grafo sparso, ma cosa ci aspettiamo dal diametro? Chiaramente a priori è difficile fare ipotesi. Come se non bastasse il grafo di facebook ha una struttura piena di comunità che si intersecano tra di loro. Possiamo pensare alle comunità come un gruppo di utenti che si che si conoscono tra di loro, ad esempio compagni di classe o colleghi di lavoro. Questa zona di grafo sarà molto densa, ma può succedere che una singola persona conosca anche persone di un'altra comunità. Questo ci fa intendere che sarà necessario definire formalmente il concetto di comunità e che nell'analisi di questo grafo sarò necessario prestare molta attenzione a queste strutture.

Rappresentazioni di un grafo e matrice laplaciana

Per semplicità assumiamo $V \subset \mathbb{N}$ di cardinalità finita.
Un grafo si può rappresentare fondamentalmente in due modi: con le liste di adiacenza oppure con la matrice di adiacenza. Il primo metodo consiste nel creare un numero di liste pari al numero di nodi del grafo, la lista i-esima contiene tutti i nodi che sono collegati tramite un arco ad nodo i-esimo. Tralasceremo questo tipo di struttura dati per memorizzare un grafo anche se il software allegato utilizza in parte questo approccio. L'altra possibilità è memorizzare il grafo attraverso la matrice di adiacenza che definiamo nel seguente modo:

$$\{H\}_{i,j} = \begin{cases} 1 & \text{se } (i,j) \in E \\ 0 & \text{altrimenti} \end{cases}$$

Definita la matrice di adiacenza possiamo definire la matrice laplaciana di un grafo come segue

$$L = D - H$$

Dove H è la matrice di adiacenza e D è una matrice diagonale con
$\{D\}_{i,i} = deg(i)$

Osservazione 1: non è difficile dimostrare che la martice L è semidefinita positiva. Infatti l'unica verifica da fare è $X \cdot L \cdot X^t \geq 0$ ma questa segue immediatamente osservando che dalla definizione $L = D - H$. A questo punto basta separare il prodotto $X \cdot L \cdot X^t = XD \cdot X^t - X \cdot H \cdot X^t$.
Si potrebbe osservare che se X è un vettore della base canonica quella è una quantità positiva (ovvio dalla definizione di D e H), altrimenti è comunque combinazione lineare di vettori della base canonica. Quindi $X \cdot L \cdot X^t \geq 0$

Osservazione 2: il vettore $e = (1, 1, ..., 1)^t$ è nel nucleo di L

Teorema 1
Dato un grafo $G = (V, E)$, allora $dim(ker(L))$ è il numero di componenti connesse del grafo

Corollario
Se il grafo $G = (V, E)$ è connesso allora $ker(L) = span(e)$

Diamo ora una serie di idee su come poter procedere e trovare la massima componente connessa del grafo per poi fare osservazioni sugli autovalori della matrice laplaciana.

Massima componente connessa e visita BFS

E' possibile trovare in $O(n)$ la massima componente connessa di in grafo, lo si fa utilizzando l'approccio con le liste di adiacenza e lanciando visite BFS. Una visita BFS è un algoritmo che visita tutti i nodi del grafo a partire da un nodo fissato. Ad esempio se si lancia la visita BFS dal nodo v, in una lista vengono memorizzati tutti i nodi adiacenti a v, ad esempio $(v_1, v_2, ..., v_k)$, si considera visitato il nodo v e si va avanti con il nodo v_1: lo considero visitato e inserisco nella lista i suoi nodi adiacenti se non sono già stati visitati (ad esempio non inserirò nuovamente v). Quindi la lista sarà diventata $(v_1, v_2, ..., v_k, w_1, ..., w_h)$ dove w_i sono nodi adiacenti a v_1. E avanti così..
Quindi non si visita mai due volte lo stesso nodo. Posso quindi contare quanti nodi ho visitato a partire dal nodo v, poi rilancio la visita BFS a partire da un nodo non visitato prima. Su quest'idea si basa la parte scritta in C nel software allegato. Quindi, d'ora in avanti considereremo solo grafi connessi.

Un'euristica per la stima del diametro

Modificando opportunamente l'algoritmo della visita BFS è possibile, a partire da un nodo v, ottenere l'insieme dei nodi che distano almeno k da v. D'ora in avanti chiameremo questo insieme

$$B_k(v) = \{w \in V | d(v, w) \leq k\}$$

Quindi considereremo $B_k(v) \setminus B_{k-1}(v)$ i vertici che hanno distanza esattamente k da v. Lanciamo la visita BFS da un v fissato (scelto a caso) e cerchiamo il k massimo tale che $B_k(v) \setminus B_{k-1}(v) \neq \emptyset$. Chiamiamo questo numero eccentricità del vertice v. Segue che il diametro è l'eccentricità massima del grafo. Quindi, per quanto detto, l'eccentricità di un nodo si può trovare in $O(n)$, quindi calcolare il diametro di un grafo costerebbe $O(n^2)$ che è troppo. Allora consideriamo il seguente algoritmo.

- Calcolo l'eccentricità di v
- Calcolo l'eccentricità di un nodo w che ha la massima distanza da v
L'eccentricità di w è un numero più grande o al limite uguale a quella di v. Quindi proseguo così per $m << O(n^2)$, ad esempio con $m = log(n)$.
Se il grafo è casuale si può dimostrare che questo algoritmo restituisce un'ottima stima del diametro. Se il grafo è di una rete sociale, sotto molti aspetti è simile ad un grafo casuale e l'algoritmo funziona.

Connettività algrabrica di un grafo

Dato un grafo G, la sua connettività algebrica λ_2 è il secondo autovalore più piccolo della matrice laplaciana.

Teoreme di Fiedler

Vale la seguente

$$\lambda_2 = \min \frac{2n \sum_{(u,v)\in E} (x_u - x_v)^2}{\sum_{u\in V} \sum_{v\in V} (x_u - x_v)^2}$$

Dove $x \in \mathbb{R}^n$ è un vettore non costante (nelle componenti) e x_v è la sua coordinata $v-$esina con $v \in V$

conseguenze: Anche se non lo si vede subito non è difficile intuire che la connettività algebrica diminuisce con il numero di nodi e aumenta con il numero di archi.

Una possibile definizione di comunità

Non sono tutt'ora note definizioni di comunità per un grafo. Per darne una definizione a livello intuitivo possiamo pensare al grafo di facebook. I primi esempi di comunità sono: i concittadini, i colleghi di lavoro, il gruppo di amici, la famiglia ed altri simili. Quindi un insieme di individui che si conoscono quasi tutti tra loro. Possiamo immaginare una comunità come una zona densa del grafo, per quanto appena detto, se consideriamo il sottografo $G' = (V', E')$, con $V' \subset V$ e $E' = E \cap (V' \times V')$, questo forma una comunità se la sua connettività algebrica è abbastanza grande. Non essendoci una buona definizione di comunità lasciamo l'ambiguità sull' 'abbastanza', in teoria questo sarà una sorta di margine, ma analizzeremo nel concreto questo aspetto.

Isomorfismo tra grafi

Diremo che un grafo G è isomorfo ad un grafo G' se esiste una rinumerazione dei vertici di G' che lo rende uguale a G.
Si può dimostrare che il problema dello stabilire se due sottografi di un grafo G sono isomorfi è NP-completo. Probabilmente il problema dello stabilire se due grafi sono isomorfi è NP (ma in generale non in P o in NP-completo) ma non ci

sono attualmente dimostrazioni di questo fatto. Ad ogni modo è un'operazione computazionalmente difficile stabilire se due grafi sono isomorfi. Si può dimostrare che la connettività algebrica è un'invariante per isomorifismo, quindi abbiamo un primo criterio per stabilire se due grafi non sono isomorfi.

Affrontiamo ora il problema numerico della ricerca della connettività algebrica di un grafo. Il problema si riconduce ad un problema agli autovalori e agli autovettori. Mostreremo due metodi per fare questo calcolo.

Osservazioni sulla struttura della matrice laplaciana

Abbiamo già osservato che la matrice laplaciana è definita positiva. Per il teorema spettrale abbiamo la decomposizione $L = Q \cdot D \cdot Q^t$ con Q unitaria e D diagonale. Quindi

$$L = \lambda_1 v_1 v_1^t + \lambda_2 v_2 v_2^t + ... + \lambda_n v_n v_n^t$$

dove $0 \leq \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$, daltronde abbiamo visto che $\lambda_1 = 0$, quindi $0 < \lambda_2 \leq \lambda_2 \leq ... \leq \lambda_n$, possiamo dunque riscrivere

$$L = \lambda_2 v_2 v_2^t + ... + \lambda_n v_n v_n^t$$

Considero ora la matrice elementare di Householder P tale che $Pe = e_1$, dove $e = (1, 1, ..., 1)$. Trasformo per similitudine la matrice L

$$B = PLP = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & C \end{array}\right]$$

Dove C è una matrice definita positiva con gli stessi autovalori di L tranne lo 0 e con ordine $n - 1$.

A questo punto il problema diventa quello del trovare il più piccolo autovalore di C, qui proporremo due metodi per farlo: l'algoritmo di Lanczos e il metodo delle potenze inverse.

Considerazioni preliminari

La matrice di Householder esplicitamente sarà della forma

$$P = I - \beta u \cdot u^t$$

dove

$$\beta = \frac{2}{\|u\|^2}$$

Allora imponiamo che $P \cdot e = e_1$, ovvero

$$Pe = \begin{pmatrix} \pm\sqrt{n} \\ 0 \\ \cdots \\ 0 \end{pmatrix} = e - \beta u \cdot u^t \cdot e$$

Se ne ricava che

$$u = e - \begin{pmatrix} \pm\sqrt{n} \\ 0 \\ \cdots \\ 0 \end{pmatrix} = \begin{pmatrix} 1+\sqrt{n} \\ 1 \\ \cdots \\ 1 \end{pmatrix}$$

Dove il segno è stato opportunamente scelto per evitare errori di cancellazione. Quindi dalla condizione su β troviamo

$$\beta = \frac{1}{n+\sqrt{n}}$$

Lo scopo di tutto ciò è osservare che il prodotto matrice vettore non è computazionalmente costosto.

$$P \cdot y = y - \frac{\sum y_i \cdot y_1 \cdot \sqrt{n}}{n+\sqrt{n}} \begin{pmatrix} 1+\sqrt{n} \\ 1 \\ \cdots \\ 1 \end{pmatrix}$$

Si era detto inoltre che è necessario calcolare gli autovalori estremi della matrice C, se questa matrice è sparsa sarà preferibile usare metodi iterativi che prevedono moltiplicazione matrice-vettore, per far questo osserviamo che se vogliamo calcolare $C \cdot y$ allora possiamo considerare $x = [0, y]$, allora

$$PLP \cdot x = \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & C \end{array}\right) \cdot \begin{pmatrix} 0 \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ C \cdot y \end{pmatrix}$$

Se il grafo è sparso allora L è una matrice sparsa, simmetrica e semidefinita positiva. Il prodotto per la matrice L costa poco, come anche il prodotto per la matrice di Householder, quindi i metodi iterativi che prevedono il prodotto matrice vettore sono evidentemente convenienti.

Metodo di Lanczos

Diamo una breve descrizione dell'algoritmo di Lanczos senza dimostrare nulla, ci soffermeremo solo sugli aspetti che riguardano il costo computazionale.
L'algoritmo di Lanczos serve per stimare gli autovalori all'estremo dello spettro di una matrice.
Sia A una matrice di ordine n di cui vogliamo al'autovalore più grande e l'autovalore più piccolo.

Breve descrizione dell'algoritmo

v_1 scelto a caso
$v_0 = 0$
$\beta_1 = 0$

$j = 1, .., k$
$w_j = A \cdot v_j - \beta_j v_{j-1}$
$\alpha_j = w_j \cdot v_j$
$\beta_{j+1} = \|w_j\|$
$v_{j+1} = v_j / b_{j+1}$

L'obbiettivo è memorizzare la successione degli α_i e β_i in modo da avere la matrice seguente

$$T = \begin{pmatrix} \alpha_1 & \beta_2 & & & & 0 \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & \ddots & & \\ & & \ddots & \ddots & \beta_{m-1} & \\ & & & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ 0 & & & & \beta_m & \alpha_m \end{pmatrix}$$

Allora vale che gli autovalori agli estremi dello spettro della matrice T sono un buona stima per gli autovalori della matrice A. Le condizioni di arresto possono scegliersi in vari modi.
Ad ogni modo trovare gli autovalori di una matrice tridiagonale è computazionalmente più conveniente (ad esempio è possibile utilizzare le successioni di Sturm). Quindi abbiamo una stima dell'autovalore più piccolo e pià grande di A.

Osserviamo che l'algoritmo deve esser applicato sulla matrice C, quindi il prodotto matrice vettore lo si deve fare come mostrato prima.

Metodo delle potenze inverse

Un'altra possibilità per trovare la molteplicità algebrica è usare il metodo delle potenze inverse, questo è computazionalmente meno conveniente dato che è necessario risolvere ad ogni passo un sistema lineare, daltronde usando questo metodo oltre che la molteplicità algebrica otteniamo anche l'autovettore corrispondente. Questo autovettore è noto come "vettore di Fiedler" ed ha importantissime proprietà strettamente collegate al concetto di comunità, ma questi aspetti saranno trattati più avanti.

Breve descrizione dell'algoritmo

Se A è una matrice non singolare diagonalizzabile, allora consideriamo la successione seguente

$$\begin{cases} A \cdot u_k = t_{k-1} \\ t_k = \dfrac{1}{\beta_k} u_k \end{cases}$$

Qundi per ogni passo bisogna risolvere $A \cdot u_k = t_{k-1}$ e β_k è uno scalare di normalizzazione, ovvero è tale che $\|t_k\| = 1$. Si ha che

$$\beta_k \to \frac{1}{\lambda_n}$$

λ_n è l'autovalore più piccolo di A e t_k è l'autovettore corrispondente della matrice A^{-1} (e quindi di A).

Il costo computazionale è dovuto alla risoluzione di un sistema lineare, ma se la matrice è definita positiva ed è simmetrica allora è possibile risolvere il sistema con il metodo del gradiente coniugato.

Metodo del gradiente coniugato

Data una matrice A definita positiva allora è possibile risolvere un sistema lineare $Ax = b$ nel seguente modo

Breve descrizione dell'algoritmo

Inizializziamo le seguenti variabili

x_0 scelto a caso
$r = b - A \cdot x_0$
$w = -r$
$z = A \cdot w$
$a = (r^t \cdot w)/(w^t \cdot z)$
$x = x_0 + a \cdot w$
$B = 0$

E iteriamo
$B = (r^t \cdot z)/(w^t \cdot z)$
$w = -r + B \cdot w$
$z = A \cdot w$
$a = (r^t \cdot w)/(w^t \cdot z)$
$x = x + a \cdot w$

La variabile r è il resto, quindi se cerchiamo una soluzione accurata dobbiamo fermarci quando $r < \epsilon$ e il vettore x converge alla soluzione del sistema.
Nel caso esaminato l'algoritmo deve esser lanciato sulla matrice C e questa è simmetrica e definita positiva.

Osserviamo che anche qui abbiamo grossi vantaggi se il prodotto matrice vettore costa poco.

Ricostruzione del vettore di Fiedler

Il metodo delle potenze inverse deve esser usato su C dato che chiaramente L non è invertibile, daltronde se si vuole trovare il vettore di Fiedler bisogna trovare l'autovettore relativo alla connettività algebrica rispetto alla matrice L.

$$B = PLP = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & C \end{array}\right]$$

Daltronde se con il metodo delle potenze abbiamo trovato v tale che $C \cdot v = \lambda_2 v$ allora consideriamo il vettore $w = [0, v]$, avremo che

$$PLP \cdot w = \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & C \end{array}\right) \cdot \begin{pmatrix} 0 \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ \lambda_2 \cdot v \end{pmatrix} = \lambda_2 w$$

Quindi in conclusione abbiamo

$$PLP \cdot w = \lambda_2 w$$

Moltiplicando a sinistra per P

$$LP \cdot w = \lambda_2 P w$$

Quindi il vettore di Fiedler sarà $V_f = P \cdot w$

Connettività tra due nodi e algoritmo di ricerca amici

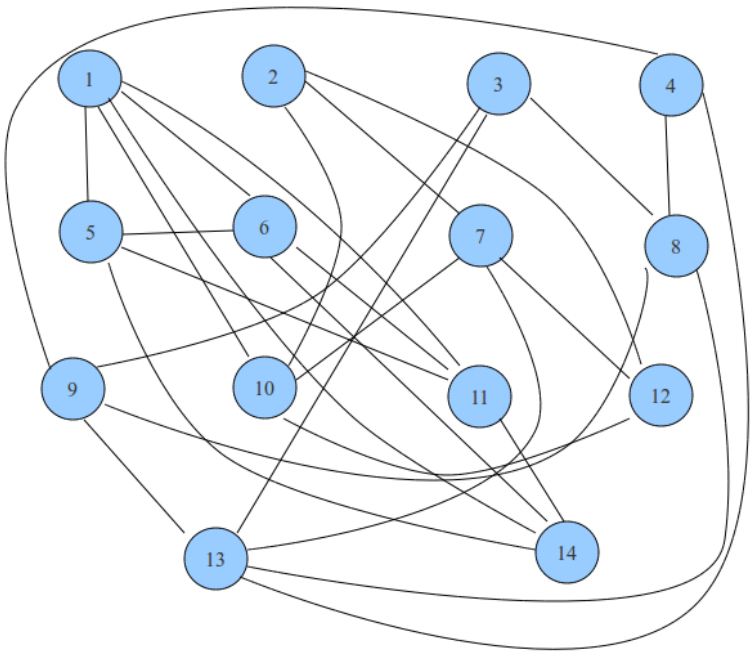
Per esprimere bene il concetto di connettività tra due nodi è preferibile partire da un esempio.
Supponiamo che un nostro amico ci presenti suo cugino, un certo "Mario" di cui non ricordiamo il cognome. Se questa persona ci è abbastanza simpatica certamente per mantenere i rapporti la cercheremo su facebook, daltronde conosciamo solo il nome quindi è ragionevole pensare che ciò sia un problema dato che in Italia esistono migliaia di "Mario", quindi facebook deve usare un algoritmo intelligente per la ricerca degli amici, ovvero deve mettere in cima ai risultati le persone che si chiamano "Mario" ma che sono più collegate a noi ed effettivamente questo è quello che succede.

Premessa
Non è detto che Facebook usi realmente questo algoritmo e non è dato saperlo dato che il codice è protetto, daltronde il risultato è lostesso.

Consideriamo il grafo di facebook e supponiamo di aver trovato il vettore di Fiedler del grafo. Dunque associamo ad ogni nodo i un valore $V_f(i)$, ovvero quanto vale l'i-esima componente del vettore di Fiedler.
Allora due nodi i e j sono tanto più connessi quanto più piccola è la differenza $|V_f(i) - V_f(j)|$.

Quandi quando su facebook cercherò "Mario" mi compariranno tra i primi risultati gli utenti che si chiamano "Mario" e che hanno il valore $V_f(j)$ vicino al mio $V_f(i)$.

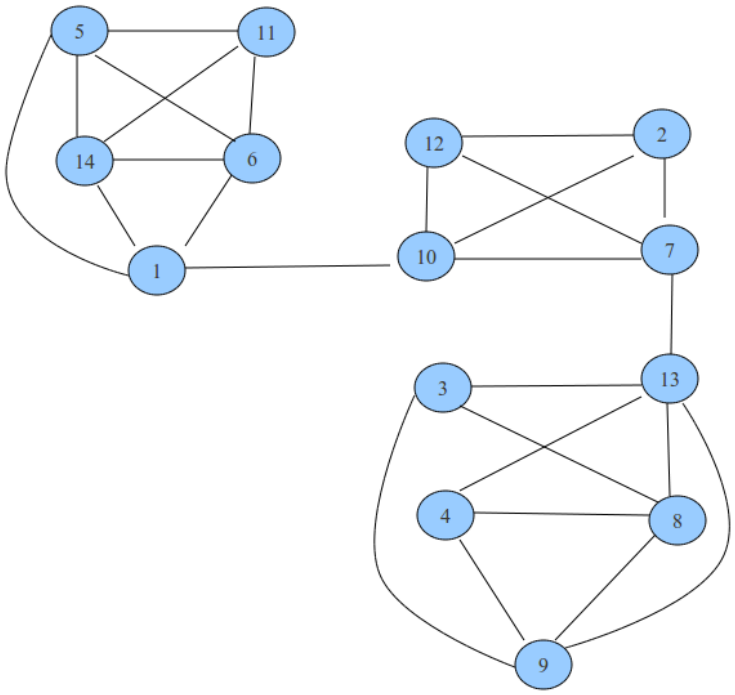
Per render chiaro il concetto facciamo un esempio, consideriamo il grafo seguente



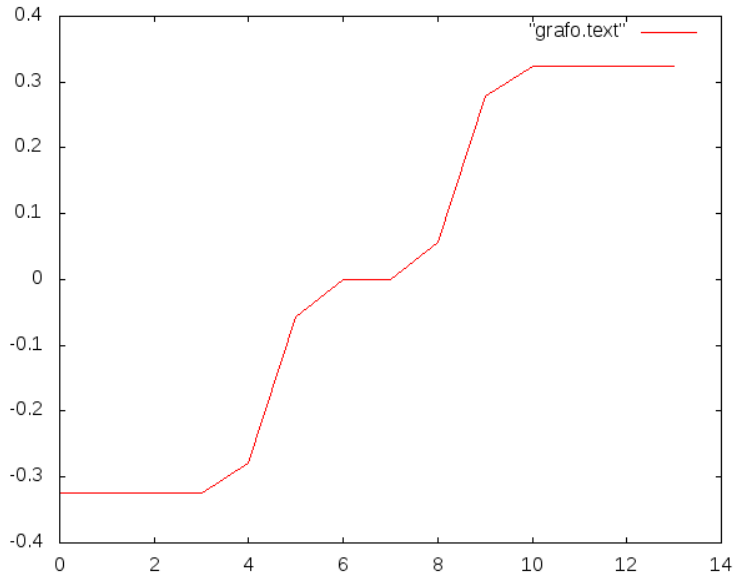
A vederlo così non è chiaro quali siano i nodi connessi meglio ad esempio al nodo 5, daltronde possiamo calcolare il vettore di Fielder e otteniamo:

Nodo	Valutazione sul vettore di Fielder
1	-0.27827907964688847
2	2.77555756156289135E-017
3	0.32375975259407974
4	0.32375975259407980
5	-0.32375975259407980
6	-0.32375975259407985
7	5.72646857300521533E-002
8	0.32375975259407974
9	0.32375975259407980
10	-5.72646857300520909E-002
11	-0.32375975259407980
12	3.46944695195361419E-017
13	0.27827907964688836
14	-0.32375975259407991

Per quanto detto prima ne deduciamo che i nodi ben connessi con 5 sono 11, 6, 14 e 1.
Daltronde riusciamo anche ad individuare altri nodi vicini, ad esempio 12, 2, 10 e 7 oppure 13, 3, 4 e 9. Infatti possiamo ridisegnare il tutto come:



A questo punto è evidente quali nodi sono vicini e quali lontani ma tale costruzione è stata possibile grazie al vettore di Fielder.
Possiamo serenamente ritornare a parlare del concetto di comunità, ora la possibile definizione proposta prima è comprensibile. Le comunità nel grafo disegnato sopra sono evidenti, abbiamo tre comunità, si trova che i sottografi hanno una connettività algebrica grande.
Infatti la connettività algebrica del grafo è 0.14047661138478115, mentre la connettività algebrica del sottografo $\{1, 14, 5, 6, 11\}$ è 3, quindi la definizione di comunità è sensata. Quindi per individuare le comunità possiamo ordinare il vettore di Fielder e vedere se ha dei comportamente ad L, ovvero se ha dei salti bruschi, ad esempio il questo caso disegnando il grafico del vettore di Fielder ordinato vediamo subito che ci sono tre comunità dato che ci sono due salti.



Quindi se il grafico è una linea retta c'è un'unica comunità (il grafico è molto denso), altrimenti fissato un margine, per ogni salto, ovvero

$$V_f(x+1) - V_f(x) > \epsilon_0$$

avremo una comunità.

In definitiva possiamo caratterizzare le comunità nel seguente modo: fissiamo una soglia ϵ e chiameremo comunità tutti i sottografi di G' tali che per ogni v e w vertici di G' abbiamo $|V_f(v) - V_f(w)| < \epsilon$.

Osserviamo inoltre che possiamo trovare anche le comunità a cui appartiene un singolo nodo v, per far questo ci basta considerare il sottografo costiutito dai nodi collegati a v e ripetere il ragionamento fatto per il caso generale.

Automorfismo canonico dato dal vettore di Fiedler

Consideriamo il grafo scritto prima, la matrice di adiacenza è la seguente

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Questa matrice non sembra avere particolari proprietà ed effettivamente non è ben chiara la rappresentazione del grafo, daltronde è possibile rinominare i vertici seguendo l'ordine del vettore di Fileder, ovvero ordiniamo i vertici

$$v_1, ..., v_n$$

tale che

$$V_f(v_i) \leq V_f(v_{i+1}) \text{ per ogni } i$$

Mettendo in ordine i vertici secondo la valutazione abbiamo

$$14, 6, 11, 5, 1, 10, 2, 12, 7, 13, 3, 8, 9, 4$$

Quindi consideriamo l'automorfismo seguente

$$\phi : G \to G$$
$$14 \to 1$$
$$6 \to 2$$
$$11 \to 3$$
$$5 \to 4$$
$$1 \to 5$$
$$10 \to 6$$
$$2 \to 7$$
$$12 \to 8$$
$$7 \to 9$$
$$13 \to 10$$
$$3 \to 11$$
$$8 \to 12$$
$$9 \to 13$$

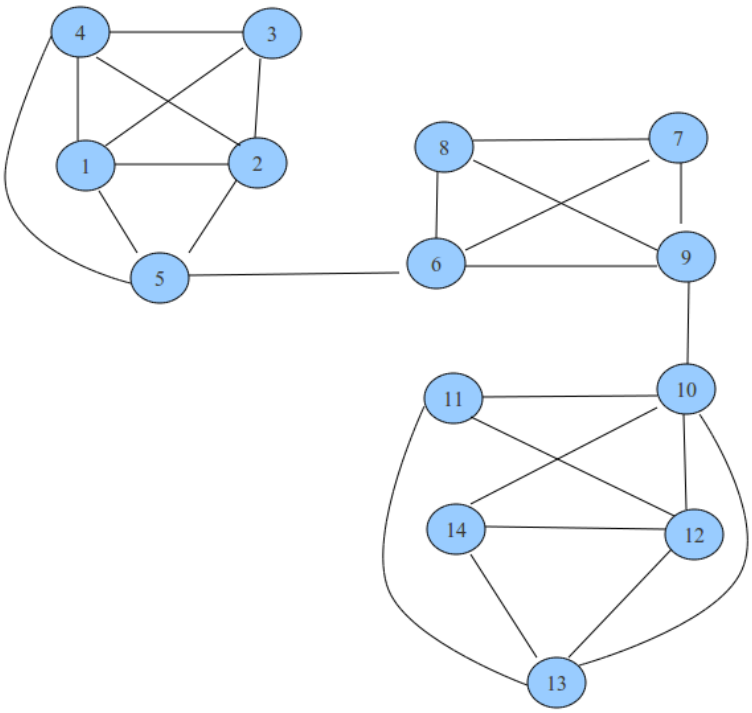
$$4 \to 14$$

Con questa rinumerazione dei vertici proviamo a riscrivere la matrice di adiacenza

$$H = \begin{pmatrix}
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0
\end{pmatrix}$$

E' evidente che è preferibile questa rappresentazione dato che la matrice è quasi diagonale a blocchi e le comunità sono evidenti già dalla matrice di adiacenza (ogni blocco è una comunità).

Infatti con questa rinumerazione è immediato disegnare il grafo in questo modo.

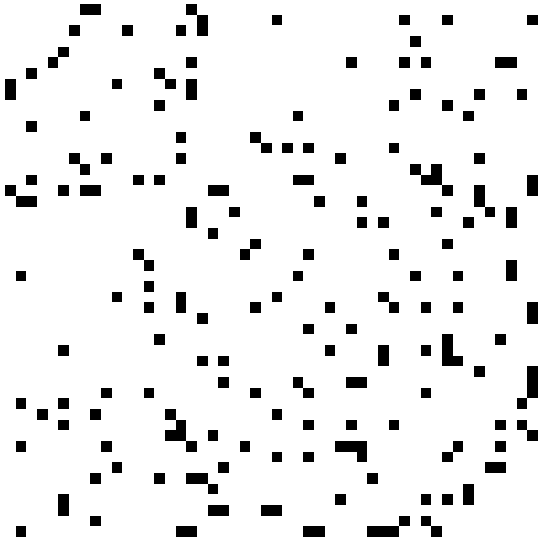


In genere non si è così fortunati ma quello che succede è che con l'automorfimo canonico (quello che segue l'ordine del vettore di Fielder) avremo un addensamento diagonale.

Esempi su grafi casuali

L'interesse di tutto ciò che si è fatto sin ora è sui grafi di reti sociali, daltronde è interessante dare un'idea di cosa succede noi grafi casuali, ovvero consideriamo n nodi e la probabilità che tra due nodi ci sia un arco è p.

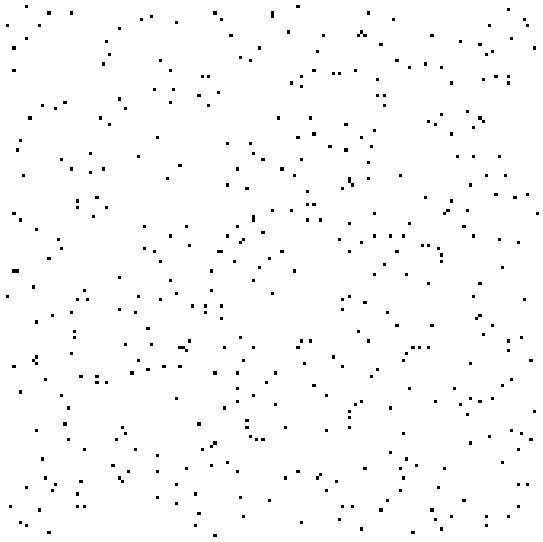
Nel software allegato c'è una subroutine " graph plot " che realizza un'immagine che rappresenta la matrice di adiacenza. Per un grafo con $n = 50$ e $p = 0.1$ otteniamo il grafo



Con l'isomorfismo canonico

Succede esattamente quello che si è detto prima, non siamo in un caso fortunato dove ci sono evidenti comunità (è un grafo casuale), daltronde abbiamo l'addensamento diagonale della matrice di adiacenza e risciamo a individuare "zone dense", ovvero comunità lungo la diagonale.

Mostriamo un altro esempio: per rendere il grafo più sparso e vedere cosa succede per il crescere dei nodi prendiamo $n = 200$ e $p = 0.01$



Con l'isomorfismo canonico

Questi esempi sono stati fatti solo per mostrare il comportamento dell'isomorfismo canonico che tende a trasformare la matrice di adiacensa in una matrice molto densa sulla diagonale dove si troveranno tutte le comunità.
Sottolineamo ancora che l'interesse di questi algoritmi è solo per grafi di reti sociali dove ha senso cercare le comunità, daltronde i grafi di reti sociali sono molto grandi e l'immagine non può esser chiara su un documento pdf, quindi sono stati presi grafi casuali per mostrare il comportamento di addensamento diagonale.

Progetti futuri

Un approccio del tipo divide et impera per la ricerca dell'automorfismo canonico

In questo lavoro è stato mostrato come ottenere il vettore di Fielder, si è anche posta particolare attenzione al caso in cui il grafo sia sparso implementando algoritmi rapidi per matrici sparse.
Tutto è stato pensato in riferimento ai grafi di reti sociali come facebook, twitter e altri social network.
Pultroppo per grafi esatremamente grandi (il grafo di facebook ha mezzo miliardo di nodi) questi algoritmi non sono applicabili neppure usando supercomputer.
Allora l'idea per rendere più veloce la costruzione dell'automorfismo canonico è quella delle bipartizioni successive.
Ovvero se abbiamo un grafo con i nodi $v_1, ..., v_n$ cerchiamo due sottoinsiemi disgiunti di vertici U e W tali che $U \cup W = V$ e inoltre $V_f(u) \leq V_f(w)$ per ogni $u \in U$ e $w \in W$.
Chiaramente supponiamo di non conoscere il vettore di Fiedler, quindi sembrerebbe non esser chiaro come si possa ottenere una simile bipartizione, daltronde c'è un algoritmo basato sul metodo montecarlo per permette di conoscere i segni delle componenti di un autovettore. Ovvero se

$$Lv = \lambda_2 v$$

allora l'algorimo mi permette di stabilire il segno dell'i-esima componente $v(i)$. Questo non univocamente determinato dato che se v è un autovettore allora anche $-v$ lo è e ha lo stesso autovalore daltronde il numero di componenti positive/negative sono invarianti, quindi la bipartizione è unica.
Per esprimere meglio il concetto, se $v = (1, -1, 1, 1, -2)$, sia che consideriamo v che $-v$ abbiamo determinato univocamente una bipartizione. Osserviamo inoltre che calcolare solo la molteplicità algebrica non è costoso se il grafo è sparso (si può ad esempio usare Lanczos). Quindi l'idea è di procedere per bipartizioni successive sino ad ottenere sottografi con molteplicità algebrica soddisfacente.

L'algoritmo ricorda molto il mergesort e si basa esattamente sulla stessa idea.

References

[1] Miroslav Fiedler : Algebraic connectivity of Graphs, Czechoslovak Mathematical Journal: 23 (98), 1973.

[2] Miroslav Fiedler: Laplacian of graphs and algebraic connectivity, Combinatorics and Graph Theory 25:57-70, 1989

[3] Dario Bini, Milvio Capovani, Ornella Menchi : Metodi numerici per l'algebra lineare, *Zanichelli*