Implementazione di due algoritmi per l'aggiornamento efficiente dell'indice di centralità di Katz

Gabriel Antonio Videtta (654839)

14 aprile 2025

Sommario

In questo progetto implementiamo e sperimentiamo la teoria e gli algoritmi proposti da [1] per aggiornare in modo accurato l'indice di centralità di Katz di un grafo non orientato semplice a cui vengono rimossi degli archi o dei nodi.

1 Trattazione della teoria

Ricordiamo brevemente che un grafo (senza archi ripetuti) è una coppia di insiemi $G = (V, E)$ con $E \subseteq V \times V$ e $V = \{1, \dots, n\}$ per qualche $n \in \mathbb{N}_{>0}$. Gli elementi dell'insieme V sono detti *nodi*, mentre quelli dell'insieme E sono detti *archi*.

Un grafo è detto *non orientato* se la relazione E è simmetrica, ovverosia se vale $(i, j) \in E \iff (j, i) \in E$, mentre è detto *orientato* se non lo è. Un grafo è detto *semplice* se è non orientato, non ha archi ripetuti e non ammette lacci, ovverosia se $(i, i) \notin E$ per ogni i. D'ora in avanti ci riferiremo ai grafi semplici come grafi, se non diversamente specificato.

Nel caso di un grafo non orientato, si può studiare il grafo G anche come una coppia (V, E) dove gli elementi di E sono insiemi di due elementi, invece che coppie, dove $\{i, j\}$ fa le veci di ciò che precedentemente erano (i, j) e (j, i). Per questo motivo ci riferiremo a un arco che collega i e j come $\{i, j\}$ e scriveremo $i \sim j$ per indicare che $\{i, j\}$ è un arco del grafo G. Se $i \sim j$, i nodi i e j sono detti *adiacenti*.

Un grafo è rappresentato operativamente tramite la propria *matrice di adiacenza* $A \in \mathbb{R}^{n \times n}$, definita componente per componente come

$$A_{ij} = \begin{cases} 1 & \text{se } i \sim j, \\ 0 & \text{altrimenti.} \end{cases}$$

Dalla simmetria di E discende immediatamente anche quella di A, e quindi $A^T = A$.

Un *cammino di lunghezza r dal nodo i al nodo j* è una sequenza ordinata di $r + 1$ nodi $i_0 = i$, i_1, ..., $i_r = j$ dove i_k e i_{k+1} sono adiacenti per ogni $k = 0$, 1,

..., $r-1$. Un cammino è detto *elementare*[1] se ogni nodo è toccato al più una volta.

Un grafo è detto *connesso* se dati due nodi esiste sempre un cammino che li collega. D'ora in avanti considereremo soltanto grafi connessi.

Osservazione 1. *Assumere di star lavorando su grafi connessi non indebolisce i risultati di [1], dal momento che è sempre possibile partizionare un grafo nelle sue componenti connesse e dunque permutare la matrice di adiacenza A in una matrice diagonale a blocchi, dove ogni blocco è la matrice di adiacenza di una singola componente connessa.*

1.1 Definizioni preliminari e notazione

Una nozione intermedia tra cammino e cammino elementare è necessaria per la trattazione che faremo della teoria in questo progetto:

Definizione 1. *Sia G un grafo connesso e sia $r \in \mathbb{N}_{>0}$. Si dice* first-passage walk (FPW) *di lunghezza r da i a j un cammino $i_0 = i$, i_1, ..., i_{r-1}, $i_r = j$ dove $i_k \neq j$ per ogni $k = 0$, ..., $r-1$. Definiamo FPW anche il cammino di lunghezza 0 in cui compare solo j. Equivalentemente un FPW è un cammino da i a j nel quale j è incontrato una singola volta, ovverosia alla fine.*

Prima di definire una nozione di distanza tra i nodi del grafo G, ricordiamo un classico risultato della teoria dei grafi, facilmente dimostrabile tramite induzione.

Lemma 1. *Sia A la matrice di adiacenza di G e sia $r \in \mathbb{N}$. Allora l'entrata (i,j)-esima di A^r rappresenta il numero di cammini di lunghezza r da i a j.*

Osservazione 2. *Dal momento che stiamo lavorando su grafi connessi, se i e j sono due nodi distinti, grazie al Lemma 1, deve esistere $r \in \mathbb{N}_{>0}$ tale per cui l'entrata (i,j)-esima di A^r è non nulla.*

Definizione 2. *Siano i e j due nodi. Allora si definisce la* distanza (geodesica) *tra i e j come*

$$\mathrm{dist}(i,j) = \operatorname*{argmin}_{r \in \mathbb{N}} \{(A^r)_{ij} > 0\}. \tag{1}$$

Equivalentemente, $\mathrm{dist}(i,j)$ è la lunghezza del cammino più corto da i a j, per il Lemma 1.

Nel corso di questo progetto, in analogia a [1], scriveremo $\mathbf{1} \in \mathbb{R}^n$ per riferirci al vettore composto da soli uno, $\mathbf{0} \in \mathbb{R}^n$ per riferirci al vettore nullo, $I \in \mathbb{R}^{n \times n}$ per riferirci alla matrice identità e $\mathbf{e}_k \in \mathbb{R}^n$ per riferirci alla k-esima colonna I. Scriveremo $\mathbf{a}_k$ per riferirci alla k-esima colonna di una matrice A e a_k per riferirci al k-esimo elemento di un vettore $\mathbf{a}$.

[1] In italiano è spesso ignorata la differenza tra cammino e cammino elementare e ci si riferisce molto spesso a entrambi semplicemente come cammino. In inglese la differenza è fortunatamente più sottile: al primo ci si riferisce come *walk*, e al secondo come *path*.

1.2 L'indice di centralità di Katz e modellazione del problema

Osservazione 3. *Dal Lemma 1 segue facilmente che l'i-esima coordinata del vettore $A^r\mathbf{1}$ rappresenta il numero di cammini di lunghezza r che partono da i e che terminano in un qualsiasi nodo.*

Definizione 3. *Sia $\alpha \in (0, 1/\rho(A))$[2], l'indice di centralità di Katz in α di A è il vettore*

$$\mathbf{x} := (I + \alpha A + \alpha^2 A^2 + \cdots)\mathbf{1} = (I - \alpha A)^{-1}\mathbf{1}.$$

Equivalentemente, $\mathbf{x}$ è l'unica soluzione del sistema lineare $(I - \alpha A)\mathbf{x} = \mathbf{1}$.

L'obiettivo di [1] è quello di sviluppare degli algoritmi efficienti per aggiornare il vettore di Katz $\mathbf{x}$ senza dover nuovamente risolvere il sistema lineare $(I - \alpha A)\mathbf{x} = \mathbf{1}$. Per fare ciò si sono considerati due casi: quello in cui viene rimosso un arco e quello in cui viene rimosso un nodo. Nel primo caso, per rimuovere l'arco (i, j) si procede annullando le entrate A_{ij} e A_{ji} nella matrice di adiacenza di G. Nel secondo caso, invece di rimuovere il nodo i, lo si "scollega" da ogni altro nodo, mantenendo così la taglia della matrice di adiacenza A invariata; operativamente si annullano dunque tutte le entrate A_{it} e A_{ti} al variare di t tra i nodi.

Osservazione 4. *Le due operazioni di modellazione ora discusse corrispondono a due modifiche di rango 2:*

$$A - [\mathbf{e}_i, \mathbf{e}_j][\mathbf{e}_j, \mathbf{e}_i]^T \quad e \quad A - [\mathbf{e}_i, \mathbf{a}_i][\mathbf{a}_i, \mathbf{e}_i]^T.$$

Se $\mathcal{S} \neq \emptyset$ è un insieme di vertici o di archi, indichiamo con $\mathbf{c}_r^{\mathcal{S}} \in \mathbb{R}^n$ il vettore la cui i-esima componente rappresenta il numero di cammini di lunghezza r passanti per almeno un elemento di $\mathcal{S}$. Il seguente risultato segue facilmente dal Lemma 1 e dall'Osservazione 3.

Lemma 2. *Sia $A_{\mathcal{S}}$ la matrice di adiacenza ottenuta eliminando dal grafo gli elementi di $\mathcal{S}$, allora*

$$\mathbf{c}_r^{\mathcal{S}} = (A^r - A_{\mathcal{S}}^r)\mathbf{1}. \tag{2}$$

Uno dei principali obiettivi di [1] è quello di trovare una scrittura alternativa a (2) per ricavare $A_{\mathcal{S}}^r$ e aggiornare così l'indice di Katz. Infatti, se $\mathbf{x}^{\mathcal{S}}$ è l'indice di Katz ottenuto rimuovendo gli elementi di $\mathcal{S}$, allora[3]

$$\mathbf{x} - \mathbf{x}^{\mathcal{S}} = (I + \alpha(A - A_{\mathcal{S}}) + \alpha^2(A^2 - A_{\mathcal{S}}^2) + \cdots)\mathbf{1} = \sum_{r=0}^{\infty} \alpha^r \mathbf{c}_r^{\mathcal{S}}. \tag{3}$$

1.3 Riscritture di $\mathbf{c}_r^{\mathcal{S}}$ togliendo degli archi

L'osservazione chiave fatta in [1] riguarda la decomposizione di un cammino passante per $\mathcal{S} = \{e\}$ con $e \in E$ in tre parti: una parte iniziale in cui e non è ancora stato visitato; la visita di e; e infine un cammino senza vincoli su tutto il grafo. Da questa osservazione segue la prossima proposizione.

[2] Il limite superiore $1/\rho(A)$ è necessario affinché la serie converga a $(I - \alpha A)^{-1}$.

[3] Dal momento che $0 \leq A_{\mathcal{S}} \leq A$, $\rho(A_{\mathcal{S}}) \leq \rho(A)$ [3] e quindi la convergenza di $\sum_{r=0}^{\infty} \alpha^r A^r$ implica quella di $\sum_{r=0}^{\infty} \alpha^r A_{\mathcal{S}}^r$. Pertanto le scritture di (3) sono ben definite.

Proposizione 1. *Siano $e = \{u, v\} \in E$, $\mathcal{E} = \{e\}$ e $r > 0$. Allora*

$$\mathbf{c}_r^{\mathcal{E}} = \left(\sum_{k=0}^{r-1} (A_{\mathcal{E}})^k (\mathbf{e}_u \mathbf{e}_v^T + \mathbf{e}_v \mathbf{e}_u^T) A^{r-k-1} \right) \mathbf{1}. \tag{4}$$

Una dimostrazione della proposizione si trova in [1, Proposition 1] e si basa sull'osservazione appena fatta.

Osservazione 5. *Dal momento che $0 \leq \mathbf{e}_i^T (A_{\mathcal{E}})^k e_u \leq \mathbf{e}_i^T A^k e_u$, se $\mathbf{e}_i^T A^k e_u = 0$ allora $\mathbf{e}_i^T (A_{\mathcal{E}})^k e_u = 0$. Il termine $\mathbf{e}_i^T A^k e_u$ coincide con $(A^k)_{iu}$, che, per la Definizione 2, è necessariamente nullo quando $k < \ell_u := \mathrm{dist}(i, u)$. Analogamente $\mathbf{e}_i^T A^k e_v$ è nullo quando $k < \ell_v := \mathrm{dist}(i, v)$.*

Combinando le tre osservazioni, $\mathbf{e}_i^T (A_{\mathcal{E}})^k e_u$ si annulla con $k < \ell_u$ e $\mathbf{e}_i^T (A_{\mathcal{E}})^k e_v$ si annulla con $k < \ell_v$.

Sfruttando l'Osservazione 5 ed espandendo i termini, l'equazione (4) si scrive termine a termine come

$$(\mathbf{c}_r^{\mathcal{E}})_i = \mathbf{e}_i^T \left(\sum_{k=\ell_u}^{r-1} (A_{\mathcal{E}})^k \mathbf{e}_u \mathbf{e}_v^T A^{r-k-1} + \sum_{k=\ell_v}^{r-1} (A_{\mathcal{E}})^k \mathbf{e}_v \mathbf{e}_u^T A^{r-k-1} \right) \mathbf{1}. \tag{5}$$

Se invece $\mathcal{E}$ è un insieme di più archi, posto che $\mathcal{E} \ni e = \{u_e, v_e\}$, allora l'equazione (4) si generalizza all'equazione

$$\mathbf{c}_r^{\mathcal{E}} = \sum_{e \in \mathcal{E}} \left(\sum_{k=0}^{r-1} (A_{\mathcal{E}})^k (\mathbf{e}_{u_e} \mathbf{e}_{v_e}^T + \mathbf{e}_{v_e} \mathbf{e}_{u_e}^T) A^{r-k-1} \right) \mathbf{1}, \tag{6}$$

dove si ricorda che $A_{\mathcal{E}}$ rappresenta la matrice di adiacenza ottenuta eliminando dal grafo G gli archi di $\mathcal{E}$.

Tramite la definizione di $A_{\mathcal{E}}$ e un cambio dell'ordine delle sommatorie si può a sua volta riscrivere l'eq. (6) come

$$\mathbf{c}_r^{\mathcal{E}} = \sum_{k=0}^{r-1} (A_{\mathcal{E}})^k A^{r-k} \mathbf{1} - \sum_{k=0}^{r-1} (A_{\mathcal{E}})^{k+1} A^{r-k-1} \mathbf{1}. \tag{7}$$

1.4 Riscritture di $\mathbf{c}_r^{\mathcal{S}}$ togliendo dei nodi

Abbiamo modellato l'eliminazione di un nodo w come una modifica della matrice di adiacenza in cui annulliamo ogni entrata relativa a w. Possiamo dunque utilizzare l'equazione (6) con $\mathcal{E} = \{\{w, u\} \in E\}$ e $A_{\mathcal{N}} = A_{\mathcal{E}}$ per ricavare

$$\begin{aligned} \mathbf{c}_r^{\{w\}} = \mathbf{c}_r^{\mathcal{E}} &= \sum_{k=0}^{r-1} \left((A_{\mathcal{E}})^k (\mathbf{e}_w \mathbf{a}_w^T + \mathbf{a}_w \mathbf{e}_w^T) A^{r-k-1} \right) \mathbf{1} \\ &= \left(\mathbf{e}_w^T A^r \mathbf{1} \right) \mathbf{e}_w + \sum_{k=0}^{r-1} (\mathbf{e}_w^T A^{r-k-1} \mathbf{1}) (A_{\mathcal{N}})^k A \mathbf{e}_w. \end{aligned} \tag{8}$$

dove si è usato che $\sum_{v \in V} A_{vw} \mathbf{e}_v = \mathbf{a}_w$, che $\mathbf{e}_w^T A^{r-k-1} \mathbf{1}$ è uno scalare e che

$$(A_{\mathcal{N}})^k \mathbf{e}_w = \begin{cases} \mathbf{e}_w & \text{se } k = 0, \\ 0 & \text{se } k > 0. \end{cases}$$

Il vettore $(A_{\mathcal{N}})^k A\mathbf{e}_w$ conta[4] il numero di first-passage walks (FPWs) da un qualsiasi nodo verso w, e saperlo calcolare efficientemente ci permetterà di sviluppare un algoritmo per determinare $\mathbf{c}_r^{\{w\}}$, e di conseguenza il nuovo indice di centralità di Katz.

Indichiamo con $\mathbf{q}_k \in \mathbb{R}^n$ il vettore contenente alla i-esima coordinata il numero di FPWs con origine il nodo i e destinazione w. Per quanto appena osservato vale che

$$(\mathbf{q}_k)_i = \begin{cases} 0 & \text{se } 0 \leq k < \text{dist}(i,w), \\ \mathbf{e}_i^T (A_{\mathcal{N}})^{k-1} A\mathbf{e}_w & \text{altrimenti,} \end{cases} \qquad (i \neq w).$$

Se invece $i = w$, per $k = 0$ il termine $(\mathbf{q}_k)_w$ vale 1, mentre vale 0 per $k > 0$ (vd. Definizione 1). Sostituendo $\mathbf{q}_k$ in (8), si ottiene immediatamente il seguente risultato.

Proposizione 2. *Sia $\mathbf{c}_r^{\{w\}}$ il vettore le cui entrate contano il numero di cammini di lunghezza r terminanti in un qualsiasi nodo dopo aver visitato w almeno una volta. Allora*

$$\mathbf{c}_r^{\{w\}} = \sum_{k=0}^{r} (\mathbf{e}_w^T A^{r-k} \mathbf{1}) \mathbf{q}_k. \tag{9}$$

In linea con quanto detto prima, e soprattutto alla luce della Proposizione 2, per rendere più efficiente l'aggiornamento dell'indice di Katz è sufficiente dunque trovare una riscrittura di $\mathbf{q}_k$.

Tuttavia non esiste un analogo di (6) per l'eliminazione di più nodi (cfr. [1, p. 8]), e si rende necessaria l'introduzione di una generalizzazione del concetto di FPW per ovviare al problema.

Definizione 4. *Sia $G = (V, E)$ un grafo. Siano $w \in V$, $\mathcal{F} \subseteq V \setminus \{w\}$, $i \in V$ e $r \in \mathbb{N}$. Un $\mathcal{F}$-avoiding first-passage walk da i a w di lunghezza r è un FPW*

$$i_0 = i, i_1, \ldots, i_{r-1}, i_r = w$$

tale per cui $i_k \notin \mathcal{F}$ per ogni $k = 0$, ..., $r - 1$. Si include tra i $\mathcal{F}$-avoiding FPW anche il cammino di lunghezza 0 diretto verso w.

Per $\mathcal{F} = \emptyset$, un $\mathcal{F}$-avoiding FPW coincide con un normale FPW.

Indichiamo con $\mathbf{q}_k^{\mathcal{F}} \in \mathbb{R}^n$ il vettore contenente alla i-esima coordinata il numero di $\mathcal{F}$-avoiding FPWs con origine il nodo i e destinazione w. Analogamente a com'è descritto $\mathbf{q}_k$, vale che

$$(\mathbf{q}_k^{\mathcal{F}})_i = \begin{cases} 0 & \text{se } 0 \leq k < \text{dist}(i,w), \\ \mathbf{e}_i^T (A_{\mathcal{F} \cup \{w\}})^{k-1} A_{\mathcal{F}} \mathbf{e}_w & \text{altrimenti,} \end{cases} \qquad (i \neq w).$$

Se invece $i = w$, per $k = 0$ il termine $(\mathbf{q}_k^{\mathcal{F}})_w$ vale 1, mentre vale 0 per $k > 0$ (vd. Definizione 4). Osserviamo che per $\mathcal{F} = \emptyset$, $\mathbf{q}_k^{\mathcal{F}}$ si riduce in effetti a $\mathbf{q}_k$.

Utilizzando gli $\mathcal{F}$-avoiding FPWs si generalizza anche la Proposizione 2.

[4] Infatti il numero di FPWs di lunghezza k con sorgente i è dato da $\sum_{j \in V} \mathbf{e}_i^T (A_{\mathcal{N}})^{k-1} \mathbf{e}_j \mathbf{e}_j^T A\mathbf{e}_w = \mathbf{e}_i^T (A_{\mathcal{N}})^{k-1} \left(\sum_{j \in V} \mathbf{e}_j \mathbf{e}_j^T \right) A\mathbf{e}_w = \mathbf{e}_i^T (A_{\mathcal{N}})^{k-1} A\mathbf{e}_w$.

Proposizione 3. *Sia $\mathbf{c}_r^{\mathcal{N}}$ il vettore le cui entrate contano il numero di cammini di lunghezza r terminanti in un qualsiasi nodo dopo aver visitato almeno un nodo di $\mathcal{N}$. Allora*

$$(\mathbf{c}_r^{\mathcal{N}})_i = \begin{cases} \sum_{w\in\mathcal{N}} \sum_{k=1}^{r} \left[\mathbf{q}_k^{\mathcal{F}_w}(w)\right]_i \left[A^{r-k}\mathbf{1}\right]_w & \text{se } i \notin \mathcal{N}, \\ (A^r\mathbf{1})_i & \text{altrimenti,} \end{cases}$$

dove $\mathcal{F}_w = \mathcal{N}\setminus\{w\}$ e $\mathbf{q}_k^{\mathcal{F}_w}(w)$ è il vettore contenente alla i-esima coordinata il numero di $\mathcal{F}_w$-avoiding FPWs di lunghezza k con sorgente i e destinazione w.

Una dimostrazione di questa proposizione si trova in [1, Proposition 4] e sfrutta dei risultati preliminari, ottenuti partizionando additivamente A in tre matrici: A^{in}, relativa al grafo in cui gli archi collegano solo elementi di $\mathcal{N}$, A^{out}, relativa al grafo in cui gli elementi di $\mathcal{N}$ si collegano solo a elementi non appartenenti a $\mathcal{N}$, e $A_{\mathcal{N}}$.

Un modo di calcolare efficientemente $\mathbf{q}_{k+1}^{\mathcal{F}}$ ricorsivamente con $\mathbf{q}_k^{\mathcal{F}}$ è dato dalla seguente proposizione.

Proposizione 4. *Sia $\mathcal{N} = \mathcal{F}\cup\{w\}$, dove $w \in V$ and $\mathcal{F} \subseteq V\setminus\{w\}$. Allora vale che*

$$\mathbf{q}_{k+1}^{\mathcal{F}} = A\mathbf{q}_k^{\mathcal{F}} - \sum_{j\in\mathcal{N}} (\mathbf{e}_j^T A\mathbf{q}_k^{\mathcal{F}})\mathbf{e}_j, \quad \text{per ogni } k = 0,1,2,\ldots.$$

La dimostrazione della Proposizione 4 si basa sulla riscrittura di $\mathbf{q}_{k+1}^{\mathcal{F}}$ come $A_{\mathcal{N}}\mathbf{q}_k^{\mathcal{F}}$ e di $A_{\mathcal{N}}$ come $A - \sum_{j\in\mathcal{N}}[\mathbf{e}_j, \mathbf{a}_j][\mathbf{a}_j, \mathbf{e}_j]^T$ (cfr. Osservazione 4, [1, Proposition 5]).

1.5 Algoritmi per l'aggiornamento dell'indice di Katz

Grazie alla riscrittura della variazione dell'indice di Katz data dell'eq. (3), possiamo adesso ricavare efficientemente nei vari casi il nuovo indice di Katz.

Nel caso in cui si tolga degli archi, mettendo insieme l'eq. (3) e l'eq. (7), si ottiene il seguente risultato.

Corollario 1. *L'indice di Katz dopo la rimozione di un insieme di archi $\mathcal{E}$ è*

$$\mathbf{x}^{\mathcal{E}} = \mathbf{x} - \sum_{\{u,v\}\in\mathcal{E}} \left[\left(\sum_{k=0}^{\infty} \alpha^{k+1} (A_{\mathcal{E}})^k \mathbf{e}_v \right) x_u + \left(\sum_{k=0}^{\infty} \alpha^{k+1} (A_{\mathcal{E}})^k \mathbf{e}_u \right) x_v \right]. \quad (10)$$

Nel caso di un solo arco $e = \{u, v\}$, la formula si riduce a

$$\mathbf{x}^{\{e\}} = \mathbf{x} - x_u \sum_{k=0}^{\infty} \alpha^{k+1} (A_{\mathcal{E}})^k \mathbf{e}_v - x_v \sum_{k=0}^{\infty} \alpha^{k+1} (A_{\mathcal{E}})^k \mathbf{e}_u. \quad (11)$$

Per costruire l'Algoritmo 1, che aggiorna l'indice di Katz dopo la rimozione di un arco, si è imposto innanzitutto per le due serie nell'eq. (11) un limite superiore $L \in \mathbb{N}$. Questo limite è scelto dinamicamente dall'algoritmo in modo tale che sia al più un limite statico $L_{max}^{\mathcal{E}} \in \mathbb{N}$ e che sia il minimo per cui la norma della differenza tra la serie per $L+1$ e quella per L sia minore di una tolleranza `tol`.

Questo corrisponde a controllare che valga

$$\alpha^{L+1}\left(x_u\frac{\|A_{\mathcal{E}}^L\mathbf{e}_v\|_2}{\|\mathbf{x}\|_2}+x_v\frac{\|A_{\mathcal{E}}^L\mathbf{e}_u\|_2}{\|\mathbf{x}\|_2}\right)<\texttt{tol}.$$

Finché questa condizione non vale e si è sotto il numero massimo di iterazioni, si reitera ogni somma dell'eq. (11).

Algoritmo 1 Aggiornamento dell'indice di Katz dopo la rimozione di un arco

Input: $A\in\mathbb{R}^{n\times n}$ matrice di adiacenza, $\mathbf{x}\in\mathbb{R}^n$ vettore di Katz, $\alpha\in(0,{}^1\!/_{\rho(A)})$, $L_{\max}^{\mathcal{E}}\in\mathbb{N}$, $\texttt{tol}\in\mathbb{R}$ tolleranza, $e=\{u,v\}$ arco da rimuovere.

Output: $\widehat{\mathbf{x}}^{(e)}\in\mathbb{R}^n$ aggiornamento approssimato, $L\in\mathbb{N}$ numero di iterazioni finale.

$\widehat{\mathbf{x}}^{\{e\}}=\mathbf{x}-\alpha*x_v*\mathbf{e}_u-\alpha*x_u*\mathbf{e}_v$;
$L=1$;
$\mathbf{s}=\alpha^2*(A*\mathbf{e}_u-\mathbf{e}_v)$;
$\mathbf{t}=\alpha^2*(A*\mathbf{e}_v-\mathbf{e}_u)$;
$\widehat{\mathbf{x}}^{\{e\}}=\widehat{\mathbf{x}}^{\{e\}}-x_v*\mathbf{s}-x_u*\mathbf{t}$;
while $\|x_v*\mathbf{s}+x_u*\mathbf{t}\|\,/\,\|\mathbf{x}\|_2>\texttt{tol}$ **and** $L<L_{\max}^{\mathcal{E}}$ **do**
  $\mathbf{s}=\alpha*(A*\mathbf{s}-s_u*\mathbf{e}_v-s_v*\mathbf{e}_u)$;
  $\mathbf{t}=\alpha*(A*\mathbf{t}-t_u*\mathbf{e}_v-t_v*\mathbf{e}_u)$;
  $\widehat{\mathbf{x}}^{\{e\}}=\widehat{\mathbf{x}}^{\{e\}}-x_v*\mathbf{s}-x_u*\mathbf{t}$;
  $L=L+1$;
end while

Per rimuovere un nodo il percorso è del tutto analogo. Mettendo insieme l'eq. (3) e la Proposizione 3, si ottiene il seguente corollario.

Corollario 2. *L'indice di Katz dopo la rimozione di un insieme di nodi $\mathcal{N}\subseteq V$ in G è tale per cui*

$$x_i-x_i^{\mathcal{N}}=\begin{cases}\sum_{w\in\mathcal{N}}\left(x_w\sum_{r=\ell_w}^{\infty}\alpha^r(\mathbf{q}_r^{\mathcal{F}_w})_i\right) & \text{se } i\notin\mathcal{N},\\ x_i-1 & \text{altrimenti.}\end{cases}\tag{12}$$

Nel caso di un singolo nodo w, l'indice di Katz diventa esattamente

$$x_i^{\{w\}}=x_i-x_w\sum_{r=1}^{\infty}\alpha^r(\mathbf{q}_r)_i.\tag{13}$$

Ancora una volta, per costruire l'Algoritmo 2, si è troncato le serie a un limite $L\in\mathbb{N}$, scelto nello stesso modo proposto per l'Algoritmo 1. Questa volta la condizione da soddisfare è però

$$\alpha^L x_w\frac{\|\mathbf{q}_L\|_2}{\|\mathbf{x}\|_2}<\texttt{tol}.$$

Algoritmo 2 Aggiornamento dell'indice di Katz dopo la rimozione di un nodo

Input: $A \in \mathbb{R}^{n\times n}$ matrice di adiacenza, $\mathbf{x} \in \mathbb{R}^n$ vettore di Katz, $\alpha \in (0, {}^1/_{\rho(A)})$, $L^{\mathcal{N}}_{\max} \in \mathbb{N}$, $\texttt{tol} \in \mathbb{R}$ tolleranza, w nodo da rimuovere.

Output: $\widehat{\mathbf{x}}^{\{w\}} \in \mathbb{R}^n$ aggiornamento approssimato, $L \in \mathbb{N}$ numero di iterazioni finale.

$\widehat{\mathbf{x}}^{\{w\}} = \mathbf{x}$;
$L = 1$;
$\mathbf{q} = \alpha * A * \mathbf{e}_w$;
$\widehat{\mathbf{x}}^{\{w\}} = \widehat{\mathbf{x}}^{\{w\}} - x_w * \mathbf{q}$;
while $x_w \|\mathbf{q}\|_2 \,/\, \|\mathbf{x}\|_2 > \texttt{tol}$ **and** $L < L^{\mathcal{N}}_{\max}$ **do**
  $\mathbf{q} = \alpha * A * \mathbf{q}$;
  $q_w = 0$;
  $\widehat{\mathbf{x}}^{\{w\}} = \widehat{\mathbf{x}}^{\{w\}} - x_w * \mathbf{q}$;
  $L = L + 1$;
end while
$(\widehat{\mathbf{x}}^{\{w\}})_w = 1$

Ricordando che il costo computazionale di un prodotto matrice-vettore Av con $v \in \mathbb{R}^n$ è $\mathcal{O}(m)$, dove m è il numero di archi di G, si ricava facilmente che l'Algoritmo 1 e l'Algoritmo 2 hanno entrambi complessità computazionale $\mathcal{O}(Lm)$ (cfr. [1, Proposition 8]).

2 Implementazione e sperimentazione in MATLAB

Presso [4] si sono implementati in `katz_edge.m` e `katz_node.m`, rispettivamente, gli Algoritmi 1 e 2. La sperimentazione è stata portata a termine tramite MATLAB R2024b su un PC fisso equipaggiato di 32 GB di RAM e di una CPU Intel i7-10700K con un clock rate di 3.80 GHz.

2.1 Illustrazione degli esperimenti

La sperimentazione segue quella di [1, Section 7], e procede considerando il grafo dei collegamenti stradali dello stato del Minnesota (`minnesota`), disponibile su [2] e a cui ci si riferirà d'ora in poi con A. Nella Tabella 1 sono riassunte le informazioni di base del grafo in questione. Si osserva subito che `minnesota` è un grafo *sparso*.

Tabella 1: Informazioni di base sul grafo `minnesota`: i numeri di nodi n, il numero di archi m, il raggio spettrale $\rho(A)$ e il numero di componenti fortemente connesse.

Grafo	n	m	$\rho(A)$	comp. fort. connesse
`minnesota`	2640	3302	≈ 3.232397	2

La sperimentazione si è divisa in due fasi e in entrambe si è utilizzato $\alpha = {}^{0.85}/_{\rho(A)}$.

Nella prima fase si è considerato l'arco $e = \{1011, 1015\}$ e si è costruita la matrice di adiacenza $A_{\mathcal{E}}$ del grafo ottenuto eliminando l'arco e. Si è dunque calcolato il nuovo vettore di Katz usando i tre seguenti metodi:

(i) Algoritmo 1 con $L^{\mathcal{E}}_{\max} = 30$ e $\texttt{tol} = 10^{-4}$.

(ii) `pcg` ($\hat{\mathbf{x}}^{(0)} = 0$): si è risolto direttamente $(I - \alpha A_{\mathcal{E}})\hat{\mathbf{x}} = \mathbf{1}$ utilizzando la funzione built-in di MATLAB `pcg` (metodo del gradiente coniugato) senza precondizionamento, con vettore iniziale $\mathbf{x}^{(0)} = 0$ e con una tolleranza relativa di 10^{-5}.

(iii) `pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{x}$): si è risolto direttamente $(I - \alpha A_{\mathcal{E}})\hat{\mathbf{x}} = \mathbf{1}$ utilizzando la funzione built-in di MATLAB `pcg` (metodo del gradiente coniugato) senza precondizionamento, con vettore iniziale $\hat{\mathbf{x}}^{(0)} = \mathbf{x}$ (il precedente vettore di Katz) e con una tolleranza relativa di 10^{-5}.

L'errore relativo tra l'indice di Katz approssimato col metodo (i) e l'indice di Katz calcolato col metodo (iii) è risultato essere

$$\frac{\|\hat{\mathbf{x}} - \mathbf{x}^{\mathcal{E}}\|}{\|\mathbf{x}^{\mathcal{E}}\|} \approx 1.56 \cdot 10^{-4}.$$

Nella Tabella 2 sono riportati i dati riguardanti il primo esperimento.

Tabella 2: Numero di iterazioni e tempi di esecuzione per approssimare $\hat{\mathbf{x}}$ con i metodi (i)-(iii) dopo la rimozione dell'arco $e = \{1011, 1015\}$. I valori rappresentano la media calcolata su 30 esecuzioni delle rispettive funzioni.

Grafo	Algoritmo 1		`pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{0}$)		`pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{x}$)	
	L	tempo (s)	iter	tempo (s)	iter	tempo (s)
`minnesota`	**7**	$\approx 5.93 \cdot 10^{-4}$	**20**	$\approx 4.71 \cdot 10^{-4}$	**10**	$\approx 2.87 \cdot 10^{-4}$

Nella seconda fase si è considerato analogamente il nodo $w = 1011$, poi rimosso. Si è costruita la matrice di adiacenza $A_{\mathcal{N}}$ e si è poi calcolato il nuovo vettore di Katz usando i tre seguenti metodi:

(i) Algoritmo 2 con $L^{\mathcal{E}}_{\max} = 30$ e $\texttt{tol} = 10^{-4}$.

(ii) `pcg` ($\hat{\mathbf{x}}^{(0)} = 0$): si è risolto direttamente $(I - \alpha A_{\mathcal{N}})\hat{\mathbf{x}} = \mathbf{1}$ utilizzando la funzione built-in di MATLAB `pcg` (metodo del gradiente coniugato) senza precondizionamento, con vettore iniziale $\mathbf{x}^{(0)} = 0$ e con una tolleranza relativa di 10^{-5}.

(iii) `pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{x}$): si è risolto direttamente $(I - \alpha A_{\mathcal{N}})\hat{\mathbf{x}} = \mathbf{1}$ utilizzando la funzione built-in di MATLAB `pcg` (metodo del gradiente coniugato) senza precondizionamento, con vettore iniziale $\hat{\mathbf{x}}^{(0)} = \mathbf{x}$ (il precedente vettore di Katz) e con una tolleranza relativa di 10^{-5}.

L'errore relativo tra l'indice di Katz approssimato col metodo (i) e l'indice di Katz calcolato col metodo (iii) è risultato essere

$$\frac{\|\hat{\mathbf{x}} - \mathbf{x}^{\mathcal{N}}\|}{\|\mathbf{x}^{\mathcal{N}}\|} \approx 1.62 \cdot 10^{-4}.$$

Nella Tabella 3 sono riportati i dati riguardanti il secondo esperimento.

Tabella 3: Numero di iterazioni e tempi di esecuzione per approssimare $\hat{\mathbf{x}}$ con i metodi (i)-(iii) dopo la rimozione del nodo $w = 1011$. I valori rappresentano la media calcolata su 30 esecuzioni delle rispettive funzioni.

Grafo	Algoritmo 2		`pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{0}$)		`pcg` ($\hat{\mathbf{x}}^{(0)} = \mathbf{x}$)	
	L	tempo (s)	iter	tempo (s)	iter	tempo (s)
`minnesota`	**9**	$\approx 3.48 \cdot 10^{-4}$	**20**	$\approx 4.97 \cdot 10^{-4}$	**10**	$\approx 3.07 \cdot 10^{-4}$

2.2 Analisi dei risultati

I risultati sono sostanzialmente coerenti con le conclusioni di [1]. Il numero di iterazioni L della Tabella 2 è inferiore ai numeri delle iterazioni eseguite con `pcg`. Analogamente succede la stessa cosa con la Tabella 3. Anche gli errori relativi indicano che gli Algoritmi 1 e 2 forniscono delle buone approssimazioni.

A prima vista, i tempi risultano però incoerenti. Nella Tabella 2, l'Algoritmo 1 è più lento di entrambe le esecuzioni di `pcg`. Analogamente nella Tabella 3, l'Algoritmo 2 risulta pressoché veloce quanto `pcg` con vettore iniziale $\hat{\mathbf{x}}^{(0)} = \mathbf{x}$.

Si può spiegare questa incoerenza in tre modi:

(i) L'implementazione di `katz_edge.m` e di `katz_node.m` è fatta in MATLAB, mentre `pcg` è una funzione built-in di MATLAB, risultando dunque sicuramente più ottimizzata.

(ii) La matrice di adiacenza di `minnesota` è molto sparsa nonché non eccessivamente grande (cfr. Tabella 1), e `pcg` è già fortemente ottimizzato per le matrici sparse.

(iii) Togliere un singolo arco o un singolo nodo potrebbe non far variare molto il ranking dato dal vettore di Katz, e questo velocizza molto le esecuzioni di `pcg` con vettore iniziale il vettore di Katz precedente. Un migliore esperimento potrebbe essere togliere sequenzialmente più archi o più nodi e comparare i risultati finali approssimati con quelli ottenuti tramite `pcg`.

Riferimenti bibliografici

[1] F. Arrigo, D. Bertaccini e A. Filippo. *Updating Katz centrality by counting walks*. 2024. arXiv: 2411.19560 [math.NA]. URL: https://arxiv.org/abs/2411.19560.

[2] D. Gleich. *Minnesota road network (with xy coordinates)*. A cura di T. Davis. 2010. URL: https://sparse.tamu.edu/Gleich/minnesota.

[3] R. A. Horn e C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[4] G. A. Videtta. *Codice per l'implementazione di due algoritmi per l'aggiornamento efficiente dell'indice di centralità di Katz*. https://github.com/hearot/katz-centrality-update. GitHub repository. 2025.