The swiss scientific social network

Vanessa Braglia

May 10, 2018

Contents

1 Introduction

A social network consists of a set of objects connected to each other by social relations. The best way to model social networks is using graphs (see an example in Figure 1): the objects (entities) are represented as nodes and the connections as edges between two different nodes. The most common example we can take is the World Wide Web (WWW) where we have web pages as nodes connected by hyperlinks, the edges.

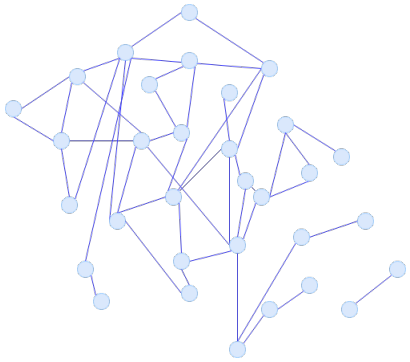

Figure 1: Example of social network graph

The goal of this project is to create a social network of computational science authors belonging to Swiss institutions and then analyze the relative graph.

Inserire qui qualche parola sul progetto e non solo sull'implementazione, in modo che la sezione non rimanga semplicemente un elenco puntato con i passi da fare.

The first step is retrieving all necessary information for the construction of the social network: names of the authors and relationships between each other. I crawled the 2015,2016 and 2017 PASC Conferences, interdisciplinary conferences which brings together research across the areas of computational science, high-performance computing, and various domain sciences, and SIAM conferences of several years, selecting the topics relevant to us (e.g. Optimization, Parallel Computing,...).

The second step is the information analysis to find out relations between institutions but also between members belonging to the same institution.

With PageRank algorithm we obtain a "ranking" of all conferences' participants: PageRank is an algorithm used by Google Search to rank websites

in their search engine results but it can be applied to any social network. In this project I use the algorithm to measure the importance of institutions' members considering the number and quality of their collaborations.

I use Graph Partitioning to "invert" the process and obtain the institutions from members collaborations: probably members of the same institution collaborate more between each other than with other institutions' representatives.

I also analyze the institutions' connectivity matrices and their structure: looking at the cliques (i.e. a sub matrix where every two distinct members collaborate with each other; this means that all entries of the sub matrix are ones) present in the matrices we can for example detect the different research areas of the institutions and the connection between them.

The results will provide an interesting picture of the different research scenarios in Switzerland and how they interact with each other.

2 Information Retrieval

...

2.1 Crawling

2.2 Parsing

3 The PageRank Algorithm

PageRank is an algorithm used by Google Search to rank websites in their search engine results; it was developed by Larry Page and Sergey Brin (the two founders of Google) in 1996 as part of their research project at Stanford University.

PageRank is used, together with other algorithms, to measure the importance of web pages and sort them by popularity in the result; it can be used in any graph to compute the importance of each node with its PageRank value. Larry Page and Sergey Brin algorithm works by counting the number and quality of links to a page to compute a value which represents its importance; more popular a page is, more likely it receives links from other websites and more likely from important websites.

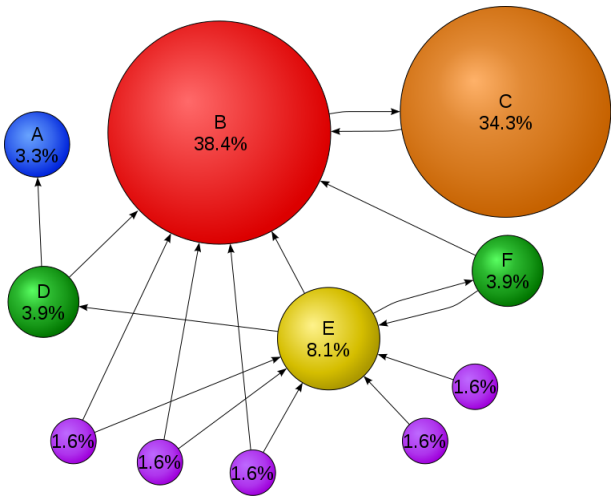

Figure 2: PageRank example

As we can see in Figure 2, node C has higher PageRank than E even if it has lower number of inner links; this is because the only inner link C has is from the most important node of the graph B so it is considered more than all the inner links of E from purple lower nodes.

Imagine surfing websites, going from one page to another: to avoid problems with pages without outer links we have two different methods to choose the next page. Our surfer can:

- randomly choose an outgoing link from the page,
- simply choose a random page from the Web.

We assume that with probability p (typically p=0.85) our surfer follows the first option and with probability $1 - p$ the second.

The probability that an infinite random surfer visits a specific website it's called its PageRank; a page will have high rank if other pages with high rank link to it.

3.1 How to compute PageRank values?

To apply PageRank, from a graph of n nodes we construct a n-by-n connectivity matrix G: $g_{ij} = 1$ if and only if node i and j are connected. The number of non-zeros in G is the total number of connections in the graph.

The row r_i represents the number of inner-links of page i, instead column c_i the number of outer-links. So we can define:

- In-degree of page i: $\quad r_i = \sum_{j=1}^{n} g_{ij}$
- Out-degree of page i: $\quad c_i = \sum_{j=1}^{n} g_{ji}$

Let p be the probability that the surfer follows a link and $1-p$ the probability that it chooses a random page, then $\delta = \frac{1-p}{n}$ is the probability that a particular random page is chosen.

We construct a transition probability matrix A where all elements are positive, smaller than one and sums of columns are equal to one; column j represents the probabilities to go from page j to all other pages. So we set:

$$a_{ij} = \begin{cases} \frac{pg_{ij}}{c_j} + \delta & \text{if } c_j \neq 0 \\ \frac{1}{n} & \text{if } c_j = 0 \end{cases}$$

If the page j has no outer links it means that column j of A assigns equal probability $\frac{1}{n}$ to all its elements. We can now compute PageRanks values solving the homogeneous linear system:

$$x = A\,x$$

The transition matrix A can be written as

$$A = pGD + ez^T$$

where D is a diagonal matrix with the reciprocals of the out-degrees

$$d_{jj} = \begin{cases} \frac{1}{n} & \text{if } c_j \neq 0 \\ 0 & \text{if } c_j = 0 \end{cases}$$

z is the vector formed by

$$z_j = \begin{cases} \delta & \text{if } c_j \neq 0 \\ \frac{1}{n} & \text{if } c_j = 0 \end{cases}$$

and e is the vector of length n with all ones.
We can write the linear system

$$(I - A)x = 0$$

as

$$(I - pGD)x = \gamma e$$

where we take $\gamma = z^T x = 1$. WHY? To conclude, we find the solution of

$$(I - pGD)x = e$$

and then rescale the solution so that $\sum_{i=1}^{n} x_i = 1$.

3.1.1 PageRank algorithm

Togliere subsection, aggiungere comando 'algorithm'.

Input: a list of authors U, the corresponding adjacency matrix G and a damping factor p
Output: list of authors sorted according to PageRank in descending order

1. Remove self collaborations $G = G - diag(G)$
2. Compute out-degree and in-degree of each node $c = sum(G, 1)$ and $r = sum(G, 2)$
3. Create diagonal matrix D with the reciprocals of the out-degrees
4. Solve $(I - pGD)x = e$
5. Normalize the result x $x = x/sum(x)$
6. Sort x in descending order

4 Graph Partitioning

Graph partitioning consists in dividing a graph $G = (V, E)$[1] cutting the smaller number of edges and obtaining sub-graphs with specific properties. For example k-way partitioning divides the graph G into k smaller components.

Inserire qualche parola di spiegazione Aggiungere immagini su Spectral/Coordinate GP

[1] A graph G with vertexes V and edges E.

4.1 Mathematical stuff...

To apply graph partitioning we need the adjacency matrix A of the graph, which is symmetric, and the Laplacian matrix L constructed such that: Fix text in following equation

$$L_{ij} = \begin{cases} d_i & if i = j \\ -1 & if i \neq j \ and \ (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

where d_i is the vertex degree of node $i \in V$. The relation between A and L is

$$L = D - A$$

where D is the diagonal matrix where d_{ii} if the vertex degree of node $i \in V$. Since L is symmetric and positive semidefinite per construction, all its eigenvalues are real and nonnegative. The first eigenvalue is always zero and the second is not equal to zero if the graph is connected[2]; we can define eigenvalues of L as follows

$$0 = \lambda_1 < \lambda_2 \leq ... \leq \lambda_n$$

The eigenvector corresponding to the first eigenvalue λ_1 is the vector of all ones and it does not provide information about the graph structure; instead the second lowest eigenvector, Fiedler vector, is used by spectral partitioning to return the smaller sub-graphs. Dividing the nodes of graph G according to the median s of the Fiedler vector $\vec{v} = (v_1, v_2, ..., v_n)$ helping obtaining two partitions V_1 and V_2:

$$\begin{cases} v_i \in V_1 \Longrightarrow v_i \leq s \\ v_i \in V_2 \Longrightarrow v_i > s \end{cases}$$

Such a partition is called the Fiedler cut and leads to two parts of G with nearly equals number of vertexes with small number of cutting edges.

4.1.1 Spectral partitioning algorithm

Togliere subsection, aggiungere comando 'algorithm'.

[2] there is a path from any node to any other node in the graph

Input: a graph $G = (V, E)$
Output: two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

1. Compute Fiedler vector of L = D - A
2. Sort the vector values
3. Put first half of the nodes in V_1
4. Put second half of the nodes in V_2
5. E_1 be the set of edges with both nodes in V_1
6. E_2 be the set of edges with both nodes in V_2

4.2 K-way Partitioning

K-way partitioning of a graph G of n nodes, consists in a division of its nodes in k disjoint subset, all with size nearly $\frac{n}{k}$.

We consider only the simple case where we want to divide the graph in k partitions and k is a power of two. The idea is just to apply the spectral partitioning $\frac{k}{2}$ times recursively. Aggiungere immagini su k-way GP

4.2.1 Recursive Spectral partitioning algorithm

Togliere subsection, aggiungere comando 'algorithm'.

Input: a graph $G = (V, E)$ and number k of desired partitions
Output: k graphs $G_1 = (V_1, E_1)$ ... $GK = (V_K, E_K)$

1. Apply spectral partitioning on G to find G_1 and G_2
2. If $\frac{k}{2} > 1$
   - 2.1 recursive partition on G_1 with $\frac{k}{2}$
   - 2.2 recursive partition on G_2 with $\frac{k}{2}$
   - 2.3 $k = \frac{k}{2}$
3. Return partitions G_1 ... G_K

5 Numerical results / Data analysis / ...

Cliques[3] ...

5.1 PageRank – forse (inclusi in R.analysis)

5.2 Graph Partitioning – forse (inclusi in R.analysis)

6 Conclusions

...

Appendix (per il codice?)

7 Bibliography

- https://en.wikipedia.org/wiki/PageRank
- Assignment1 1 Course Numerical Computing 2107/2018
- Assignment2 1 Course Numerical Computing 2107/2018

[3] Some text in a footnote.

- ”https://am.vsb.cz/export/sites/am/cs/theses/kabelikova_ing.pdf”