Hierarchical Clustering Analysis using Graph based models for Author Dissambiguation task

Carlos López Roa
me@mr3m.me
DMKM

June 24, 2016

Abstract

In this work we develop a clustering analysis based on graphs for the author disambiguation task. That is, a general framework of clustering into natural and novel approach using graphs into a task of entity resolution to exploit the inner structure this social network. Methods and details of the implementation are drawn, also experiments are carried on and the results exposed. The code developed is available in this Github repository mr3m/WebOfScience4J

1 Introduction

Hierarchical clustering analysis (HCA) seeks to build structure in a unlabelled dataset, in the form a hierarchy of cluster. It can be constructed using a dissimilarity measure between the elements to characterise, and this can be a *distance* function. From this, a hierarchy of agglomerating clusters can be drawn and can be graphically represented with a dendogram.

Graphs in the context of graph theory are representations of objects as nodes linked by edges. The relation between two nodes can be arbitrary and not uniform, leading to *multigraphs* which contain different classes of nodes and different classes of edges. A graph which represent social phenomena is often called social network.

Collaboration Networks first described by Paul Erdös consists of a graph of individuals represented as dots and they relations represented as edges connecting the dots.

Collaboration Networks can be fully defined with two sets, a set of vertices and a set of edges. This class of graphs can be proved to be *simple*[1] and *small world*[2], the later characterises the *distance* between any two points, defined as

$$d(v_i, v_j) = \arg\min_k P(v_i, v_j), \tag{1}$$

that is the minimum length path between to vertices[3], to be

$$\begin{aligned} E[d(v_i, v_j)] &= K \\ K &\propto \log N, \end{aligned} \tag{2}$$

that is the average distance between any two vertices grows as the logarithm of the number of vertices.

In some cases a collaboration can be characterised as *scale free*[4] if the presence of hubs the average distance grows as

$$K \propto \log\log N. \tag{3}$$

[1] With no self relations, all edges are undirected thus making the edges a set and not a multiset.

[2] In which most nodes are not neighbors of one another, but most nodes can be reached from every other node by a small number of hops or steps

[3] If the graph is not *connected* and there exists no path between to vertices, their distance is said to be infinite

[4] or ultra-small world

Small world graphs can be modelled as *Watss-Strogatz random graphs* with just two parameters, namely the clustering coefficient and the average node-to-node distance.

Author disambiguation in the context of entity resolution consists in finding the true relations between a pair of authors, this commonly being the binary relation : Is author A the same person as author B? Several approaches have been proposed, ranging from traditional blocking based on string or phonetical distances and statistical learning based on a manually disambiguated training set, also traditional relation entity resolution techniques have been tested with promising results.

Certainly a new approach to this task is to incorporate a social network approach, which is normally hard or impossible to compute in traditional RDMS, whereas here a novel approach is proposed using a Graph Database[5] which can reduce the complexity by a factor of n^2 that is, taking from $\mathcal{O}(n^2)$ to $\mathcal{O}(\log n)$ in computing the relations between two nodes. [?]

2 Methods

Here we took an approach of modelling the data as given but in the graph database context, that is, to approach the problem as a data modelling rather that mathematical modelling. From a stable and clear graph database model several questions can be answered, rather than sticking to a unified mathematical model based in assumptions.

For this the SQL dumps were converted into CSV format and imported into a provided 4 core, 36GB server. Different from traditional RDBMS a Graph database is queried using a language called Cypher in it we can declare, nodes, relations, properties, etc.

The procedure we followed was:

1. to create the nodes and declare the properties of each node
2. to index the properties
3. to create relations based on the properties of the nodes
4. to query to find relations.

Both an online shell and a programmable interface (API) were used for this process.

The graphical model is based solely in coauthorship relations, then only the relations of articles and authors were loaded. Using RDMS the data was denormalised from 318,591 articles and 2,328,539 authors, into 1,724,465 pairs of articles, authors, called signatures; and 475,327 unique author strings. Each signature, article and author were assigned an id, and all relations in the graph database were done using this id's.

An heterogenous graph database was done, that is using different classes of nodes and relations between them. First the data of the articles was loaded setting the properties in each node of article_id, title, authors, journal and year of the article. Then the set of authors was loaded using the unique author generated table, setting the properties of name and author_id. Then the set of signatures were loaded, this data set consists in many to many relations of the authors and articles and is ideal to model this kind of relations in a graph database.

Indexes were created in all the ids of the several nodes.

Then the relations *isAuthor* were created between a signature and a given author. Also the relations *isArticle* was created between a signature and the article contained in it. This relations are unidirectional and one to one.

Then using scripting language to manage the memory and execution time (which was in the order of milliseconds) a relation was drawn between all the authors of a given article, by the *isCoauthor* relation. By definition any author is a coauthor of himself. this was done, traversing the graph between any two pair of articles and walking the relations *isArticle* and *isAuthor* through the intermediary signature. After a tenth of a second 285,651

[5] Neo4j

relations were drawn between collaborating authors.

Then the relation *isWritten* between the articles and the unique authors was drawn traversing the graph between every article through the relation *isArticle* to the signature and then by the relation *isAuthor* back to the author. After one tenth of a second 3,445,248 relations were drawn.

This constitutes the data model and is illustrated in figure 1

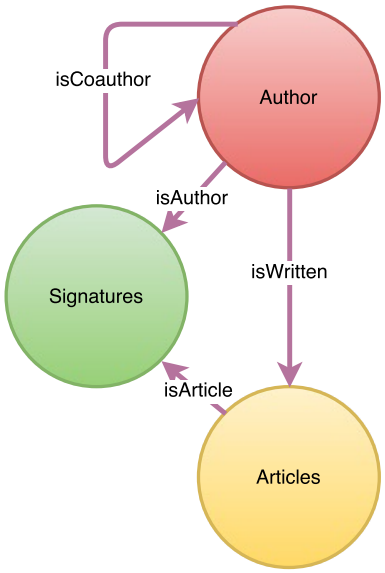

Figure 1: Representation of the node classes and their relations. In this case, the signatures are related to the authors and articles by the relations isAuthor and isArticle respectively, the articles and the authors are related by the isWritten relations and the Authors are related to themselves by the isCoauthor relation

3 Experiments

1. First a measure of the number of articles per author was drawn.

2. Then a measure of the number of coauthors per author.

4 Results

Results of the first experiment are shown in figure 2

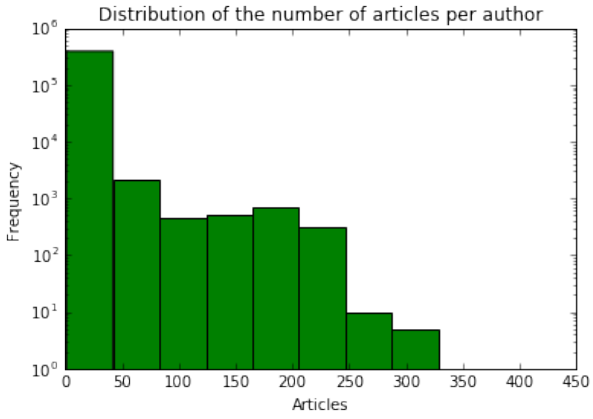

Figure 2: Distribution of the number of articles per author. Here we can see that the mayority of authors have few articles (mostly one by 42% of the authors)

Results of the second experiment are shown in figure **??**

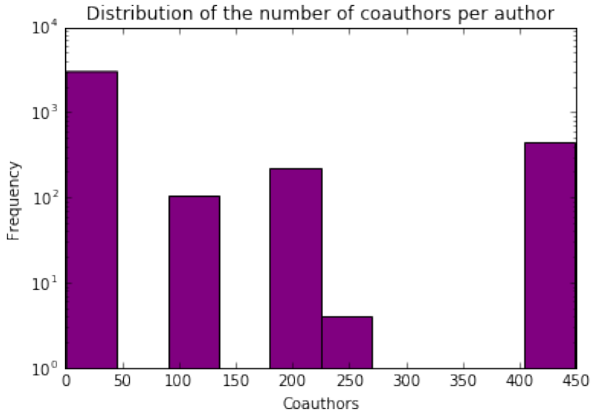

Figure 3: Distribution of the number of coauthors per author. Here we can see that the mayority of the authors only have coauthorship with themselve of few little groups and another group have form a big group of mutual coauthors, also a mayority is somehow in the middle. but this distribution is multimodal.

Seeing this we realised that the majority of authors, do not have a coauthor, leaving only

9,847 authors with coauthor, that is around 2%. We decided to study the relations between this authors solely since they form the principal component of the graph. This is composed of 187,133 nodes, that is 65% of the original coauthor relation.

The mean number of articles per author is 4.38

The mean number of coauthors per author is 75.95

A view of a 100 point sample can be seen in 4

5 Conclusions

- Using graph based data modeling we were able to propose a novel method to address the issue of Author Dissambiguation using the co-authorship network of a given literature of scientific publications.
- The developed method uses a *graph* distance function as a dissimilarity measure, which is suitable for using hierarchichal clustering.
- The heterogenous graph can actually be seen as the coexistance of several simple graphs which gives the problem both versatility and potential in the theoretical side.
- The mean distance is in factor proportional to the logarithm of the number of nodes, which defines the graph as of the *smal world* type
- The use of state of the art, graph database technologies, makes a big difference, reducing the computation complexity by several orders of magnitude.
- It is really important to index the nodes for better performance
- The APIs provided by the developer are mature and well suited for application development.
- After observing that most authors do not collaborate we were able to reduce the graph size by 35%, turning it into a more connected graph.
- The co-authorship degree is polarised into several groups, this can be due to massive collaborations with in the same lab, which yields groups or much higher degree than the median. Also a great portion of the graph is disconnected

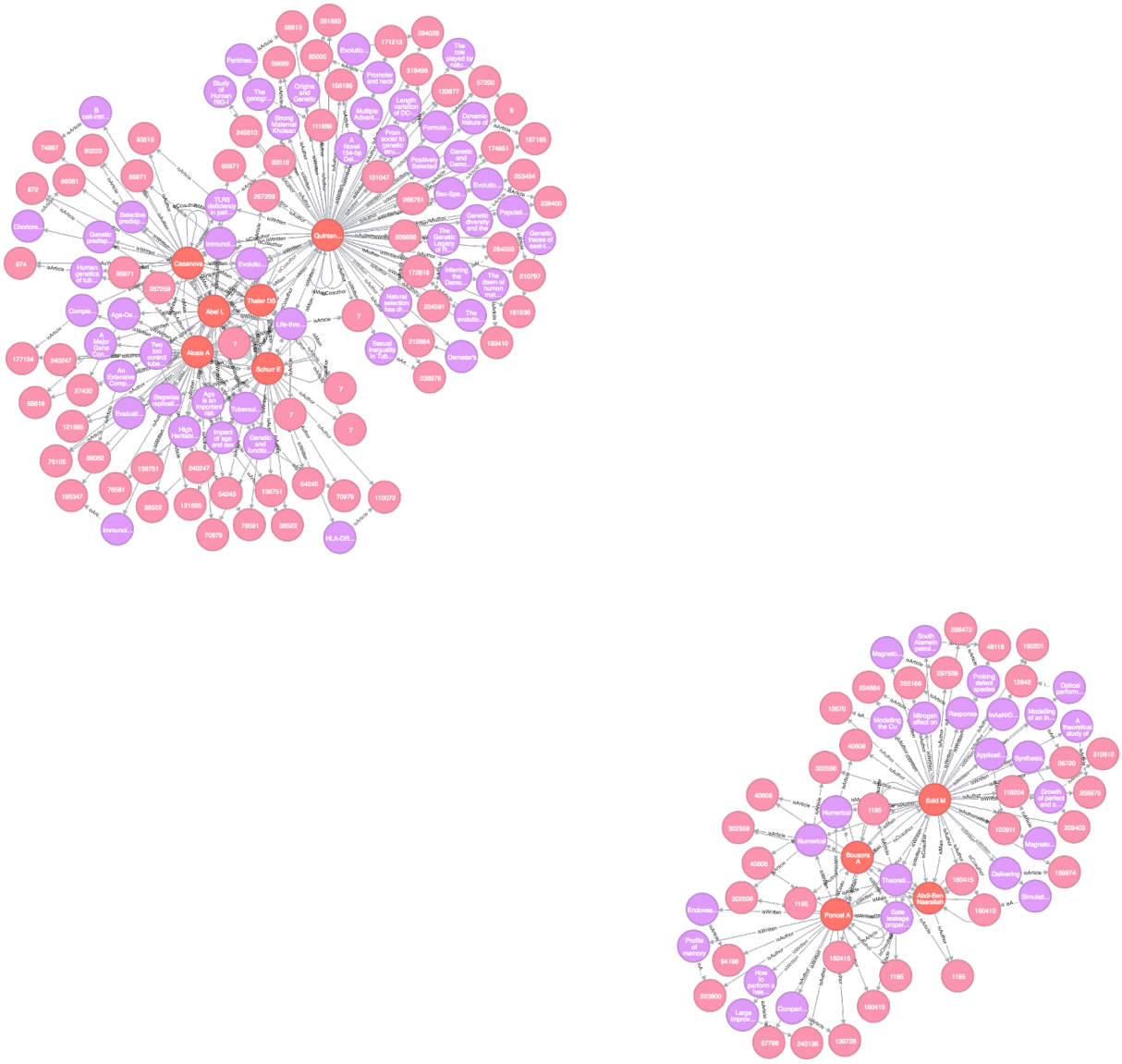

Figure 4: A 100 point sample of the heterograph, several relations are shown also each class of node is colored differently